TMN

MuTerm

ish

LHA

Bdif & Bup

# X68000 Free Software Book

グループ68k……編

Fu

MF

LZX

SEE

DC

SuperED

tsort

dedit

SRAMCLR

TwentyOne

hcommand

caps

float2p

HIOCS

FLEXDISK

DCACHE2

DE

ADDDRV

C/DINIT

X68000
フリーソフトウェア・
セレクション
5"2HD

SOFT BANK

TMN

MuTerm

ish

LHA

Bdif & Bup

# X68000 Free Software Book

グループ68k……編

Fu

MF

LZX

SEE

DC

SuperED

tsort

dedit

SRAMCLR

TwentyOne

hcommand

caps

float2p

HIOCS

FLEXDISK

DCACHE2

DE

ADDDRV

C/DINIT

SOFT BANK

# X68000 Free Software Book

グループ68k……………………編

SOFT BANK

# はじめに

## 氷山の一角としてのフリーソフトウェア本

X68000が世に出てから今年（1993年）でまる６年。巷では、新マシンも登場したようですが、この６年間のX68000の歩みを振り返ってみると、決して恵まれた環境の中で育ってきたのではない、その姿が浮かび上がってきます。

発売当初は、その優れたグラフィック表示能力に注目したソフトハウスが作ったゲームソフトが注目を浴び、X68000＝ゲームマシンなどと言われました。その後も市販ソフトの数は少なく、マニアのマシンなどと言われつづけてきました。しかし、パソコン業界全体に目を転じると、この６年間に数多のマシンが登場し、消えていきましたが、X68000はそれなりに生き残ることができました。

X68000がここまで来られたのは、ひとえにMPU68000に惚れ込み、市販ソフトの不足を自分たちのプログラム技術でカバーし、「ないものは自分たちで作ろう」という精神でやってきた人たちに支えられてきたからだといえます。そして、もう１つ見逃せないのが、X68000登場と前後して盛り上がってきたパソコン通信です。X68000のユーザ自身が、パソコン通信を通じて自分たちが作ったプログラムを公開し、寄せられた使用レポートに書かれた意見を吸収して、さらに優れたプログラムを作り上げていきました。それらのプログラムの中には、メーカのソフトウェア作りにも影響を与えるものさえ出てきています。

しかし、一方で、そのようなフリーソフトウェア（以下、フリーソフトと略す）の世界をまったく知らない人や、雑誌などでフリーソフトが紹介されていても、どうやって入手したらいいかわからず、手をこまねいている人が多いのも事実です。フリーソフトを入手できないこのような人たちは、ソフトバンクが書籍につけている読者カードの中で、「次は、フリーソフトの本を出してください」と熱心に書いてきたりしています。本書の企画の発端は、そのような読者カードが大変多いことから始まりました。

そこで本書では、X68000ユーザでありながらパソコン通信をしたことがない人に、パソコン通信とはどういうものか、そこで配布されているフリーソフトにはどんなものがあるのかなどを知ってもらうことに主眼を置いて企画されました。

また、本書ではフリーソフトとして配布されている通信ソフトや、各種通信支援プログラム、ファイラ、ツール、システム拡張プログラムなどのうちから比較的よく知られているものをディスクに収めておきました。それぞれに特徴のある、優れ

たプログラムではありますが、ここに収められているプログラムはX68000用のフリーソフトのほんの氷山の一角であることを頭の片隅に置いておいてください。数あるフリーソフトをとてもフロッピーディスク1枚に収めきれるものではありませんし、この本を手に取ってみた人は、きっと収録ソフトの数の割には分厚い本だと思われたことでしょう。でも、私たちは単なる「付録ディスク＋フリーソフトの解説」本で終わらせたくなかったのです。

本書では、まず第1章で、X68000の著名なフリーソフトの紹介、本書に収録した通信ソフトを使ったBBSへのアクセスのしかた、フリーソフトの入手方法などの記事によって、実際にパソコン通信への参加のしかたがわかるようにしました。

第2章では、フリーソフトの機能紹介に加え、フリーソフト作者の製作意図、裏話、こう使ってほしいなどというコメント（作者からの言葉）や、実際に使っている人たちによる推薦文を載せるなど、さまざまな人の声で作り上げられていくフリーソフトの世界の雰囲気が少しでも伝わるようにしてみたつもりです。

最後の第3章では、4人のフリーソフト作者を招き、座談会「ぼくらは、なぜフリーソフトを作っているのか」を行いました。作者自らに、何を期待し、なぜフリーソフトを公開しているのか、市販ソフトとは違う意図で作られているフリーソフトとのつきあい方などをたっぷり語ってもらいました。

本書の原稿執筆にあたったグループ68kは、東京の草の根BBS、MAX BBSの会員有志のグループです。私たちは、原稿執筆の他、フリーソフトの選択、最新版プログラムのディスクへの収録などの作業を担当しました。

なお、フリーソフトの収録にあたっては、以下のような基準でソフトを選ばせていただきました。

(1) パソコン通信初心者が便利に使えるプログラム（設定、操作などの難しいものはできるだけ避けました）やユーティリティ
(2) バグのない（あるいは常駐プログラムなどで、他のプログラムとの相性のいい）プログラム
(3) 2HDディスク1枚にLHAで圧縮して入れるため、プログラム自体は非常に優れたものであっても、あまりにプログラムサイズの大きいものは除外する
(4) 原稿を書く私たちが実際に使ったことのあるプログラム

これらの作品の収録のお願いをしたところ、ほとんどの作者の方が快く許可してくださいました。しかし、中には結局作者の方と連絡がとれずに収録を断念したも

のもありましたし、自分の作ったプログラムが今回の収録の基準にあわないと辞退された方もおられました。

ディスクへの収録にあたっては、できるだけネットにアップされているものと同じものを、最新版で収録するように心掛けましたが、残念ながら、いくつかのプログラムではネットにアップされているものとは違う形になってしまいました。これは、本書へのプログラムの収録を作者の方たちにお願いしたときには、その後のバージョンアップによるプログラムサイズの増大を考え、50Kバイトほどの空き容量を残しておいたのですが、諸般の事情により刊行が遅れてしまった間に、収録プログラムのサイズが大きく膨らんでしまい、2HDディスクに収まらなくなってしまったためです。

結局、何人かの作者の方にお願いをして、収録プログラムのうち、あまり使用頻度の高くないと思われるファイルの収録は見送らせてもらうことにしました。どのプログラムで、何のファイルの収録を見送ったか（あるいは、部分的にアーカイブさせていただいたか）は、それぞれのプログラムの紹介ページを見ていただくことにして、この点については、私たちの不備をお詫びしておきます。読者の方には、できれば本書に収録した通信ソフトを使って、本来の形でのプログラムを入手されることをお勧めします。

なお、パソコン通信とは違い、融通性の乏しい書籍という性格上、本書収録ソフトがバージョンアップなどによって次第に古くなっていくのは致し方ありません。最新版は、何度も申しますが、みなさん自身の手で実際にネットにアクセスして入手されることを希望します。

最後になりましたが、本書にフリーソフトの収録を許可していただいた作者の方、収録することはできませんでしたが、プログラム名だけは紹介させていただいた作者の方たちに、この場を借りてお礼申し上げます。また、本書の打ち合わせ・原稿のやりとり等はパソコン通信上でほとんど行われました。その場を提供していただいたMAX BBS、ひるま-ねっとのSys.op.の方にもお礼を申し上げます。

本書によって、パソコン通信の世界を知らないX68000ユーザに、通信の世界のおもしろさに気づいてもらい、実際にパソコン通信の世界に参加する中から優れたフリーソフトが誕生し、X68000の世界がより充実していくことを祈念してやみません。

1993年2月16日

グループ68k

contents

第1章

# X68000<br>フリーソフトウェア入門

1-1
フリーソフトウェアのすすめ

1-2
フリーソフトウェアの世界

1-3
フリーソフトウェアの定義

1-4
市販ソフトウェアとフリーソフトウェア

1-5
X68000とフリーソフトウェア

1-6
フリーソフトウェアの使用・入手

# X68000フリーソフトウェア入門

## 1-1 フリーソフトウェアのすすめ

### フリーソフトウェアの世界へようこそ！

「ちょっとこういうことをしたいんだけど、なんか適当なプログラムはないかなぁ。お店に売ってるソフトウェアで処理できるような内容じゃないし……」

「いつも使ってるプログラムなんだけど、ここがもう少しこうだったら、もっとずっと使いやすくなるのになぁ」

「○○○というソフトウェアを買ってみたんだけど、今ひとつ使いにくいなぁ。買ってみないと、どんなソフトウェアだかわからないからしようがないんだけど……」

あなたにもこんな経験がありませんか？　こんなとき、フリーソフトウェアがあなたの悩みを解決してくれるかもしれません。

フリーソフトウェアとは、現在主にパソコン通信のネットワークなどを通して配布されている、配布・使用に関する制限が市販ソフトなどと大きく異なるソフトウェアの総称です。用語の細かい定義についてはあとで述べますが、大ざっぱにいうと「電話代程度の費用で手に入れることができ、使ってみることができるソフトウェア」ということです。

一般的にいって、市販されているソフトウェアは（別にソフトウェアに限った話ではありませんが）高価なものほど機能が豊富であり、安いものは機能が足らず、また使いにくい、というイメージがあります。この論法を当てはめて考えて、ほとんどタダ同然の値段で手に入るフリーソフトウェアはとても使いものになるような代物ではない、と思われる方もいらっしゃるかもしれません。

実際にはこの考え方が大きな間違いであるということは、自分でパソコン通信を始めてフリーソフトウェアを使いはじめるようになるとすぐにわかります。もちろん、どんなものでもそうですが、フリーソフトウェアと一口に言ってもいろいろなものがあり、それこそ市販ソフトと見まがうばかりの（あるいは、それ以上の）みごとな大作ソフトウェアもあれば、小粒でもピリリッと辛い優れものプログラム、プログラミングの練習をかねて作ってみました的な簡単なプログラムまで、さまざまです。内容のほうも、コンパイラやエディタなどの開発環境や、グラフィックツールやミュージックツール、便利な小物ユーテ

ィリティから小粋なゲーム、さらにはフリーソフトウェアならではの一発ギャグプログラムまで、なんでもござれです。

本書は、シャープ㈱のパソコン、X68000シリーズをお使いのみなさんを対象として、この素晴らしいフリーソフトウェアの世界の一端をご紹介し、フリーソフトウェアを通して、みなさんのパソコンライフがよりいっそう実りあるものとなれば、という願いを込めて作られています。

また、いくつかのフリーソフトウェアをフロッピーディスクに収録して添付しておきましたので、実際に使用してみることができるようになっています。このあとの各章を読まれ、また添付のソフトウェアを使用する中でフリーソフトウェアの世界に興味を持たれたなら、あなたもぜひパソコン通信を始められ、フリーソフトウェアの世界を存分に堪能されるようお勧めします。

## 1-2

# フリーソフトウェアの世界

先ほど述べたとおり、「フリーソフトウェア」と呼ばれるソフトウェア群は現在主としてパソコン通信のネットワークを通して配布が行われています。パソコン通信では、特定のコンピュータ（ホスト局）に電話回線経由で自分のパソコンを接続することによって、そのホスト局に登録されている記事やプログラムを読み出したり、自分がホスト局にそれらを登録したりすることを主として行っています。

したがって、フリーソフトウェアは通常、まず、パソコンにモデムなどの必要な機器を接続し、パソコン通信のネットワークホスト局（BBSなどとも呼ばれる）に加入して、そのホスト局に登録してあるプログラムを自分のパソコンに転送し（この処理は「ダウンロード」と呼ばれる）、必要ならばダウンロードしたファイルを変換したあと、使用することになります。

つまり、通常フリーソフトウェアはホスト局に接続している間の電話代、プラス有料ネットの場合は、そのネットの規定の利用料金を支払うだけで使用できることになります。これらのソフトウェアは、おおむね作者または他のBBSからの転載者の手によってホスト局に転送され（この処理は「アップロード」と呼ばれる）、そのBBSの会員が自由に使用できるように登録されています。

このように、フリーソフトウェアは、まず、その流通形態からして通常店頭で販売されているソフトウェアとは異なりますが、その他にも市販のソフトウェアとはさまざまな違いがあります。その違いはあとの節で述べますが、それらの違いのおおもとになる最も基本的な点は、ソフトウェアを作る側、すなわちプログラムの作者の考え方、発想、主張、

目的、動機といったようなものが、市販のソフトウェアを製作・販売している会社とは大きく異なる点です。

市販のソフトウェアは商品であり、製作・販売をしている会社はそれによって利益を得ることを目的としています。極端な言い方になりますが、便利な機能をサポートするのも、使用者の声を反映してバージョンアップを行うのも、電話で苦情の問い合わせに対応するのも、結局はソフトウェアの売り上げを伸ばし、それによって利益を得るために行っている、ともいえます。会社は営利団体なのですから、利潤追求を行うのは当然であり、正当なことでもありますが。

一方、フリーソフトウェアはさまざまな目的のために製作・配布されます。たとえば、自分で使うために作ったものを善意で、または誰かに頼まれて公開するとか、あるいは自分のプログラミング技術を磨くために作成するとか、プログラムミスを探して欲しいために公開するとか、市販プログラムへのアンチテーゼであるとか、自己主張のためとか、ウケを狙うためとか、それこそ千差万別です。これらのさまざまな動機・目的で製作・配布されるフリーソフトウェアたちは、いわば１つの「作品」であり、作者の自己表現である、ともいえるかもしれません。

また、作者たちが求めているものも、直接的・金銭的な利益というよりも、たとえば使ってみた感想であるとか、修正点の情報であるとか、さらに改良するための提案であるとか、あるいは共通のフリーソフトウェアを使用することによる便利さや連帯感など、主に使用者との相互のコミュニケーションによって得られる、形のないものであることがほとんどだと思われます。もちろん、まったくなんの見返りも必要とせず、純粋な善意や自分のために作成している場合もあるでしょう。

したがって、これらのフリーソフトウェアを使う側も、市販のソフトウェアを使用する場合とは違った対応が求められることになります。使用しているフリーソフトウェアの作者への感謝の気持ちとか、相互のコミュニケーションの尊重とか、そういったものが大切であり、作者へのフリーソフトウェアの公開の強制とか、サポートの強要といったことがお門違いであることはおのずとご理解いただけると考えています。

なにも難しく考える必要はありません。使ってみたフリーソフトウェアに対する謝辞や感想を一言BBSに書き込むだけでもいいのです。そこからさまざまなコミュニケーションが生まれる場合もあります。こういった、いわば相互の信頼や理解に基づいて成り立っているフリーソフトウェアの世界の精神を尊重して、楽しくフリーソフトウェアとつきあっていってもらえれば、と筆者たちは考えています。

# 1-3 フリーソフトウェアの定義

ここで、「フリーソフトウェア」およびそれと類似した意味の用語について説明しておきます。なお、これらの意味はなんらかの正式な文書・規格などによって厳密に定義されていないものがほとんどで、これらのソフトウェアの製作者や、使用者、あるいは各種メディア等でおおむね「このくらいの意味だろう」という程度の定義で使われているものもあります。ここで述べている定義が唯一絶対のものとは言いきれない場合もありますのでご注意ください。

## 1-3-1 PDS

一般にパソコン通信でソフトウェアが配布されるようになった頃（あるいはさらにさかのぼれば各種研究機関でミニコン上のソフトウェアがやりとりされるようになった頃から）、それらのソフトウェアを指す意味で広く用いられていたのが「PDS」という用語です。PDSはPublic Domain Softwareの略で、作者が自作のソフトウェアに対する著作権を放棄したものや、最初から著作権が存在しないようなものを指して呼ぶことがあります。

ご存じのように、日本ではソフトウェアは著作物である、として著作権法で各種権利が保護されています。PDSという用語は、作者が「著作権を主張しないから、好きなように使ってもらってかまいませんよ」という意味で、プログラムを配布する際にソースプログラムや添付のドキュメント等に「Public Domainである」旨を書いていたことからきています。それらのソフトウェアがネットワーク等で広く配布されるようになり、PDSという用語も一般化していきました。

ところが実際には、「そのソフトウェアに関するあらゆる権利を放棄する」ということは、たとえば極端な話として、そのソフトウェアを別の誰かが「自分のものである」と主張し、それによってそのソフトウェアの配布制限や他者の利用制限、販売等を行ったとしてもなんら異議をはさめないことになりかねません。PDSという用語とそのプログラムが一般的になるにつれ、残念ながら、このような問題が起こる可能性が無視できなくなってきました。

また、現行の著作権法では、著作権はその著作物が書かれると自動的に発生し、その放棄に関しては規定されてないという見方もあり、厳密な意味でのPDSは（少なくとも日本では）存在し得ないのではないか、という議論もありました。

これらの理由から、最近では「著作権は作者が保持するが、配布・使用は自由に行ってもらってかまいません」というパターンが多くなっており、PDSという用語は実情とあわ

なくなってきたこともあってあまり用いられなくなってきています（もっとも、これらの事情を承知のうえで、あえて「PDSである」旨をわざわざ断っているプログラムも存在しないわけではありません。その辺は作者の主義の問題というわけです）。

ちなみに、プログラムに対するPDSに相当する用語として、著作権のないデータのことをPDD(Public Domain Data)ということもあります。

## 1-3-2 フリーソフトウェア

上述のような経緯から、PDSにかわる用語として使われるようになったのが、このフリーソフトウェア（フリーソフト）という言葉です。著作権・配布・使用等に関する細かい規定はプログラムごとに異なり、大抵は付属するドキュメントに書いてあるのですが、おおむね以下のような場合が多いようです。

- 著作権はプログラムの作者が保持する
- 特に断りのない限り、配布はおおむね自由。主にサポートなどの理由から、ネットワークへの転載や雑誌への掲載の場合は作者への連絡を希望している場合も多い。逆にあえて「連絡不要」と断っている場合もある
- 個人の使用に関しては自由に行ってかまわない。ただし、トラブル等が起こった場合は基本的に使用者の責任で対処する。営利目的の使用に関しては、規定がある場合が多い
- その他、使用レポートの作者への連絡希望や、プログラムの改変に関する規定などを設けている場合もある

このように、フリーソフトウェアの「フリー」とは、「配布・使用に関する制限からの自由」という意味合いが強いようです。英語の「free」には「無料の」という意味合いもあり、フリーソフトウェアは「タダで使えるソフトウェア」というふうに解釈される場合もありますが、無料（に近い状態）なのはあくまでも結果である、と考えたほうがいいかもしれません。

フリーソフトウェアという言葉自体は最近広がりだした言葉で、以前はPDSほどには一般に認知された言葉ではありませんでした。そのため、PDSという用語のかわりに用いられる類似の用語にも、オンラインソフトウェア、フリーウェアなど、さまざまなものがあります。前者は通信を媒体として配布されるプログラム全般（商用ソフトも含む場合もある）を強調して指す場合に用いられることが多いようです。後者はフリーソフトウェアと同様の意味ですが、外国で商標として登録されているらしく、日本でもあまり使用しないようにしている人もいます。

## 1-3-3 シェアウェア

フリーソフトウェアに関連してしばしば話題に出てくるものとして、「シェアウェア」といわれるソフトウェアがありますが、この2つは大きく異なったものです。シェアウェアの基本的発想は、商用ソフトウェアの流通手段としてパソコン通信などのネットワーク媒体を利用しようというものです。したがって、PDSやフリーソフトウェアなどとは異なり、著作権の保持や使用に関する注意などについてはかなり厳格に規定されていることがほとんどです。パソコンとネットワーク通信が広く普及しているアメリカでは、一般にかなり認知されている流通形態のようです。日本でもこれから広まっていくかもしれませんが、現状ではフリーソフトウェアとの境界があいまいな場合も見かけられます。

シェアウェアの配布形態には主に2つの方法があります。1つはなんらかの有用なプログラムをネットワークを通して配布し、試用してみて気に入ったならば作者（もしくは発売元の会社等）に送金する、あるいは口座などに代金を振り込むというような方法です。送金するかどうかは基本的には使用者の良心にゆだねられているわけです。

もう1つの方法は、本来のプログラムから使用日数限定などの制限をつけた縮小版・デモ版を配布し、気に入ってもらった使用者には、送金と引き替えに完全版を送る、というものです。

いずれの場合も、代金を支払う前にそのソフトウェアを試用してみることができるというメリットがあり（そういう意味で、いわゆるソフトウェアのオンラインショッピングとは違います）、使用者にとってはうれしい配布形態といえるでしょう。

## 1-3-4 GNUソフトウェア

フリーソフトウェアの考え方をより先進的に推し進めたものとして、GNUプロジェクトによるソフトウェア群、つまりGNUソフトウェアがあります。著作権を、プログラムの自由な配布を制限する目的ではなく、配布する自由を守るための積極的な権利として利用しようというのがGNUソフトウェアの考え方です。

GNUプロジェクトとは、アメリカのR.M.Stallman氏の提唱によるもので、彼の主宰する団体FSF(Free Software Foundation)が中心となって各種活動を行っています。GNUの基本的な主張は、「ソフトウェアは再利用可能な資源であり、権利の独占による使用制限とはなじまない。特にOSなどの基本的なソフトウェアは、使用・配布・改変などの自由に関する権利が使用者に保証されたものでなければならない」というもので、これに基づいて FSFではUNIX上で動作する高機能・高性能な各種コンパイラやプログラミングツールなどを無償配布しています。 FSFは、最終的にはUNIXと互換性のあるOS本体を配布することを目的とし、寄付などで運営されています。

GNUソフトウェアの権利関係については、各種GNUプログラムといっしょに配布される「GNU General Public Licence(GPL)」に詳細に記述されています。詳しくはGPLを入手して(GNUソフトウェアには添付されている) 読んでいただくとして、その趣旨は以下のようなものです。

・複製を自由に配布・販売できる
・ソースコードの入手が可能である
・変更や組み込みができる
・これらのことを使用者自身が知っている必要がある

つまり、販売してもかまわないが、その際も配布が自由であることを明記しなければならない、また配布や再利用の権利を妨げる場合は、著作権を行使して抵抗する、というわけです。これらの権利主張を、GNUでは著作権を意味するCopyrightをもじってCopyleftといっています。

このGNUソフトウェアは、本来ワークステーションなどで広く使用されているOSであるUNIX上で使用できるように作られていますが、パソコンにも移植されています。X68000では、有志の手によって早くからCコンパイラをはじめとする各種GNUソフトウェアが移植されており、その高性能さから広く利用されています。

ちなみに、GNUとはGNU's Not UNIXの略です。

# 1-4 市販ソフトウェアとフリーソフトウェア

次に、一般にパッケージとして市販されているソフトウェアとフリーソフトウェアの違いについて、いくつかの項目に分けて考えてみます。

## 1-4-1 機能

市販されている各種ソフトウェアは (特にベストセラーに名を連ねるような有名なソフトは)、おおむね高機能化路線に進みがちです。同種の他のソフトが新しい機能をサポートすると、こちらもサポートしなければ売り上げに響き、最終的には市場から淘汰されることになりかねませんので、そういう意味では自然な方向といえます。しかし、この方向が行き過ぎると、「こんな機能を一体誰が使うのだろう？」、あるいは「こんな機能まであったのか、全然知らなかった」というような一種無意味な多機能 (必然的に低速な) 鈍重ソフ

トになりがちです。

これに対し、フリーソフトウェアは逆に絞り込んだ機能で軽快な動作、という方向性のものが一般的に多いようです。これは作者が個人で作っている場合がほとんどであるため、必然的にあらゆる機能を１人で作るのは無理がある、という事情もありますが、必要な機能があればそれでいい、という作者と使用者の共通認識のもとにソフトウェアが作られているためでもあります。もし作者が欲しいと思う機能がなければ（可能ならば）自分で作るでしょうし、使用者がなんらかの機能を追加して欲しいときは作者にお願いして（作者が同意すれば）追加してもらえる、そしてどちらも必要としない機能は当然作られないというわけです。パソコン通信という双方向性のメディアの利点が生きている例です。最近ではパソコン通信を使って使用者の希望を聞き、市販ソフトのバージョンアップの参考にする例や、フリーソフトウェアとしてバージョンアップしていったソフトが市販されるといった例も出てきています。

一方では、フリーソフトウェアの中にも市販ソフトウェアも驚くような斬新・多彩な機能を持ったソフトも存在します。これはひとえに作者の努力の賜物なわけですが、作者が同時に使用者であるということが素晴らしい発想のアイデアを生み出したり、時間的制約（締切）のないプログラミング環境が段階的なバージョンアップと高機能化を可能にする、というフリーソフトウェアならではの利点が生かされているともいえます。また、この場合でも、当然上記の理由からまったく無駄に思われるような多機能性に陥ったりはしません。

また、フリーソフトウェアならではの特徴として、ソースコードが公開されているプログラムもあります。そのようなプログラムでは、現在の機能に飽きたりない人がさらなる機能拡張を施し、別のバージョンとして公開するという過程を繰り返すことによって、徐々に高機能化が進むということもよくあります。

## 1-4-2 性能

ソフトウェアの性能を語るときは、まず処理速度の速さが挙げられます。市販のソフトウェアに類似のものがすでにあるのに、新たに作られたようなフリーソフトウェアの場合、そのソフトウェアが作られた動機は（単に買いたくないからとか、作者の趣味という場合もありますが）、作者の目的とする用途に対して機能が足りなかったか、あるいは遅くて使う気にならなかったか、のどちらかである場合が大部分でしょう。

また、次の「操作性」の節でも触れますが、特に日常長時間にわたって使用するようなツールなどの場合は、使用者の操作に対するソフトのレスポンスの悪さは（使用者の性格がせっかちであるというような場合もあるかもしれませんが）かなり致命的な欠点になり得ます。遅いツールをしかたなく使ってイライラさせられた経験のある方も多いと思います。

そのため、多くのメジャーなフリーソフトウェアは、他にかえがたいような特徴のあるようなものを除けば、少なくとも使っていてストレスを感じさせない程度の処理速度は確保しているものが大部分です。そうでないソフトは、使用者が減って自然に淘汰されていくということもあります。また、最初から処理速度の高速化を目的として作成されたようなフリーソフトウェアの場合は、言うまでもなく市販のソフトより圧倒的に高速です。市販品やシステムに付属しているソフトが遅くて使いにくいため、高速化のための改造をほどこすようなプログラムも多数出回っています。

その他にソフトウェアの性能として問題にされるものとしては、システム的な容量制限が挙げられるかもしれません。たとえば、扱えるデータファイルの大きさに制限がある、というような場合です。この点についてはX68000の場合、使用されているプロセッサとOSの特徴として（16ビットマイクロプロセッサとしては）比較的制限が少ないこともあって、市販ソフトでも実メモリの許す限り、データ領域が確保できるものがほとんどです。

このことはフリーソフトウェアでも同様です。ただし、空きメモリに対する制約が少ないため、逆に最初に一気にメモリ上にデータを読み込んでしまうようなソフトが多く、実メモリサイズが小さい場合は（X68000の場合は、最低でも1Mバイトはメインメモリが存在するわけですが）そのことが制限になってしまう場合もあります。最近は市販ソフトでも2Mバイト必須のものも増えてきていますが、フリーソフトウェアの場合も、メモリはあればあるだけよく、ものによっては2M～4Mバイトくらいのメモリがないと実用にならないような大物も一部には存在します。X68000の場合、まさに「メモリは力」だといえます。

### 1-4-3 操作性

X68000用のソフトウェアの操作性に関しては、圧倒的にフリーソフトウェアに分があるといえるでしょう。フリーソフトウェアの場合、ソフトの作者はまず間違いなく、そのソフトの使用者ですし、そのソフトの仕様を最終的に決定するのも当然作者自身ですから、自分が使いづらいようなソフトを作るわけがありません。作者の余裕と能力の及ぶ範囲で、必要ならばいくらでも使いやすいようにプログラミングするはずです。

そして、いったんフリーソフトウェアとして公開すれば、そのプログラムにある程度の有用性が認められた場合、多くの一般ユーザがそのプログラムを使用することになります。各自の使用感に対する好みは十人十色ですから、当然「ここを、こういうふうに直してくれ」とか、「こうしたら、もっと使いやすくなるのでは」といった意見が使用者の数に応じて出てくることになり、それがパソコン通信を通して作者に伝えられます。作者も納得した場合は（親切な作者の場合は、自分が使わないようなことでも要望が多ければ新たに機能を付け加えるという場合すらある）、それが次のバージョンアップに反映されることにな

ります。その繰り返しでバージョンアップが続く限り改良されていくことになり、独りよがりな操作性になってしまうこともあまりありません。

当然市販ソフトでも、使用者からの意見や試作品のモニターの感想などは反映されていると思われますが、作者・使用者間のコミュニケーションの緊密さから勝負はおのずとついてしまいます。

## 1-4-4 アフターケア

ソフトウェアにはバグ、すなわちプログラムのミスによる動作不良がつきものです。プログラムの作成作業のうちのかなりの時間は、このミスを発見してプログラムを修正する作業、つまりデバッグに費やされているといっても過言ではありません。

市販ソフトウェアの場合、建て前としては商品にバグがあってはいけないわけですし、当然開発するほうでも多くの時間をさいてデバッグ作業を行っているはずですが、実際問題として巨大な市販ソフトから100%バグを取り除くことは不可能に近いといえます。作った側が予想もしなかったような操作をした場合、あるいはめったに起こらないような状況が重なったときなど、どうしてもバグが残ってしまう可能性があります（市販ソフトの中にもすぐわかるようなバグを残しているようなソフトもありますが）。

いわんやフリーソフトウェアにおいてをや、です。個人でデバッグに費やせる時間には限りがありますし、知人にバグ出しを頼むとしても限界があります。第一、趣味でやっているソフト作成で、通常おもしろくもないデバッグを延々と続けたくない、という作者側の事情もあります。フリーソフトウェアは、一般に「使っていて何か不具合があっても、作者は責任はとれませんよ」というルールで使用されていることがほとんどですので、使用者側も当然バグの修正を強制することはできません。

ちなみに、こういう書き方をするとフリーソフトウェアを使うことをためらう方が出てきそうなので、念のため言っておきますが、フリーソフトはバグが多い、と言っているわけでは決してありません。特によく使われているようなソフトでは、ディスクやデータを破壊するような致命的なバグが出ることはまずほとんどありません（誤操作などでディスクを壊してしまう可能性があるようなソフトの場合は、普通厳重な注意書きがあります）。

また、そもそもそのような事態が起こった場合は、市販ソフトでも責任がとれない（とらない）場合がほとんどですので、バグや誤操作による破壊が起こる可能性がゼロでない以上、結局のところは絶対に壊すとまずいようなデータはバックアップをとっておくしかないという考え方もあります。

いずれにしてもバグの可能性がある以上、問題はその後のサポートということになります。このアフターケアという問題についても、パソコン通信によって意見交換や頻繁な配布ができるフリーソフトウェアは一般に市販ソフトより有利だといえます。フリーソフト

ウェアでは、ソフトの公開後に使用者からバグの報告があったような場合、ただちに直して再び公開し直すということも（バグが直る場合は）簡単にできます。先の致命的なバグの話も、すぐに出るようなものは公開直後に報告されてただちに修正される（修正できない場合はできるまで、とりあえず公開したものを削除する）ことがほとんどです。

市販ソフトの場合は、一度発売してしまうと、よほどのことがない限り、修正して配布することは不可能です。次の何年先かわからないバージョンアップまで待つしかありませんし、そのときに直っているという保証はありません。第一、一般に市販ソフトは使用者側から発売元に電話なりなんなりで問い合わせをしなければ何もリアクションがないのが普通ですが、フリーソフトウェアは積極的にバグの報告があり、修正されていくというところから大きく違う点だともいえます。

ただし、フリーソフトウェアのサポートにも弱点があります。１つはほとんど個人でやっているためもあって、あらゆることに手が回るわけではないという点です。そのため、あまり致命的でなくて直しにくいバグはどうしても後回しにされる傾向もあります。また、報告されたバグが作者のところでは同じ状況でも再現されなかったといったようなこともあって、すぐには修正作業が始まらないような場合もあります。そもそも先にも書いたように修正を強制できるわけではありませんから、作者が修正やバージョンアップをする気がなければそれまでです。

## 1-5

# X68000とフリーソフトウェア

世のパソコン事情を見渡してみると、日本ではNECのPC-9800シリーズ、海外ではIBM PCおよびその互換機を中心として、CPUにIntel系のものを、OSにMS-DOS系のものを使用したパソコンが市場のかなりのシェアを占めています。パソコンのハードウェア台数が多いということは、そのマシンで使用されるソフトウェアも多いということであり、必然的に多数派マシン用のソフトは少数派のものよりも売れることになります。ソフトウェアの製作・販売をしている会社は、よく売れるほうにより力を注ぐため、少数派マシン用のソフトは発売や改版があまり行われなくなってしまうのが現実です。CPUが異なると移植が困難な場合が多いことも、その状態に拍車をかけます。その結果として少数派マシンが徐々に淘汰され、今のIntel系のCPUとMS-DOS全盛の状況になってしまったわけです(Macintoshのような例外もないわけではありませんが)。

これに対してX68000は、CPUに国産ではほぼ唯一Motorolaの68000を、OSにはオリジナルのHuman68kを採用した、上記の分類では明らかに“少数派”のマシンです。にもかかわらず、５年以上もの間淘汰されず、ある程度の販売台数を記録してきたのは、ゲームを

中心として良質のソフトをコンスタントに提供しつづけてきた一部ソフトハウスの努力と、愛着と思い入れでマシンを使いつづけて（ソフトを買いつづけて）きたユーザの努力の賜物だといえましょう。以前からX68000は「マシンの台数のわりにはソフトが売れる」とも言われ、そのためか一時急に多くのソフトハウスがX68000のソフト市場に参入し、それほど売れなかったため(ユーザ数が急に増えたわけではないので当然ですが)、「X68000にはソフトのコピーユーザが多い」という根拠の不明な中傷を受けたことまでありました。

このように少数派マシンの中で健闘を続けているX68000ですが、多数派マシンと比較すると、ソフトウェアの量的な不足はやはり大きく、特にゲーム以外の実用ソフトやユーティリティの分野では選択の幅が狭かったり、他のマシン用と同等の機能を持つソフトがまったくなかったりすることも、残念ながら少なくありません。世間の評価が「X68000はゲーム用のパソコン」と見られてしまうのも、このような状況の結果といえます。

しかしながら、たとえばゲームがやりたくてX68000を購入した人でも、年賀状を作るためにワープロとして使いたいとか、プログラミングやCG、あるいはコンピュータミュージックなど、もっとクリエイティブにパソコンを活用してみたいと思うことがあるかもしれません。もちろん、最初からそのような目的でX68000を購入する方もいるでしょう。そういう場合にX68000は、特にクリエイティブな利用という面で大きな可能性を秘めたマシンであるといえます。そのCPUである68000は比較的素直な構成でプログラミングに向いているといえますし、多彩なグラフィック表示や、FM音源、ADPCMなどの表現力豊かな周辺回路は、何もゲームのためだけにあるわけではありません。

また、そのような目的のために新たに活用しようと思った場合に、X68000は追加投資が少なくてすむマシンであるともいえます。高価なフレームバッファなどを追加購入しなくても、65536色にも及ぶ同時発色数が得られるわけです。

このような可能性を内に秘めながら、それをサポートすべきソフトウェアが弱いというのはとても残念なことです。しかし、そこであきらめてしまう必要はもうありません。X68000ユーザにはフリーソフトウェアという強い味方がいるのです。ワープロのような超大物ソフトウェアをフリーソフトウェアで完全に代替するのはちょっと苦しいですが、簡単な文章を書くくらいならフリーソフトウェアのエディタでも十分可能ですし（事実、この文章はフリーソフトウェアのエディタであるMicroEMACSとedtを用いて書かれていますし、日本語入力には、市販のFEPであるFIXERにフリーソフトウェアによる改造をほどこしたものを使用しています。ヒット率の低ささえ我慢できれば、FEPとしては本体付属のASKもあります)、プログラミングやCG、コンピュータミュージックのためのツールでも十分実用で使えるものがいくつもあります。

また、通常のコンピュータの操作環境などを改善するようなユーティリティなどの場合は、圧倒的に優れたものが存在します。フリーソフトウェアを多数導入した環境でX68000を使いはじめるようになると、もう二度とメーカから提供された純正品のみの環境では使

う気が起こらなくなってしまうことは確実です。まさに「フリーソフトウェアあっての X68000」であることを実感できることと思います。

参考として、筆者がその恩恵にあずかっている、または存在を知っているフリーソフトウェアの例をいくつか挙げてみましょう。あらかじめお断りしておきますが、ここで取り上げたプログラムはそれぞれに優れたものですが、必ずしもX68000用の多彩なフリーソフトウェアのうちの代表的なプログラムを網羅したというわけではありません。紙数（と筆者の時間的な）の都合で取り上げられなかったものがいくつもありますし、そもそも筆者がお邪魔しているパソコン通信のネットは数多くのネットのうちのごく一部ですから、筆者がその存在を知らない優れたプログラムも、おそらくは山のように存在するはずです。

なお、以下の記述でソフト名に続くかっこ内の名前は、作者や移植者の方のハンドル名(通信上のペンネームのようなもの）です。

## 1-5-1 通信関連

まず最初に、おそらくフリーソフトウェアが最も得意としている分野だと思われる、パソコン通信関連のソフトウェアから見てみましょう。パソコン通信の利用者が最も頻繁に使用すると思われるのが、ターミナルソフト（通信ソフト）を中心とした、これらのソフトウェアです。頻繁に使われるだけあって、利用者の注文やバグ報告も多く、それだけ磨かれたソフトウェアとなっています。

まず、通信に直接使用する通信ソフトとしては、今回添付ディスクに収録させていただいた**MuTerm**(はちくん作)と**TMN**(西表山猫さん／星野美季さん作)が有名でしょう。大ざっぱに特徴を述べると、高速さと手軽さのMuTerm、多機能とオープン指向のTMNという感じでしょうか。この他にも、TMNの本家にあたる**Telecom Miki**(星野美季さん作)や、強力なマクロ機能で古くからのユーザを持つ**TSTerm2**(去石さん作)など、優れたソフトがいくつも存在します。

次に、ソフトウェアの転送に用いる変換用プログラムとして、MS-DOSマシンなどでも有名な**ish**(X68000版は☆紀さん他作)があります。プログラムなどのバイナリファイルと、通信で取り扱うことのできるテキストファイル（TYPEコマンドで内容が確認できるファイル）とを相互に変換するためのプログラムです。これによって、バイナリファイル転送のための機能（プロトコル）を持たないホスト局ともプログラムの授受ができます。ほぼ同様な機能を持つプログラムとして、より高速な**mic**(みるくさん作)もあります。

また、プログラムの配布の際に関連ファイルやドキュメントなどを1つにまとめ、それを圧縮する（ファイルのサイズが小さいほうが転送時間が短くてすみ、電話代の節約になる）プログラムとして、アーカイバと呼ばれるプログラムがあります。これには、やはりMS-DOSマシンで有名な**LHA**(X68000版はともちゃん作)が挙げられます。ファイルの

種類にもよりますが、圧縮によって全体のファイルサイズが2分の1近くになることも珍しくありません。同様に圧縮・展開を行うソフトとしては、LHAの前身である**LHarc**から移植・改良された**lh**(やーさん／まりこさん作)もあります。また、展開専用で高速な**xx**(みるくさん作)もあります。

プログラムの改版を行ったとき、そのプログラムが巨大な場合は特に、小さな変更点であっても、大量のプログラム転送を行う必要が出てしまいます。これを避けるために、プログラムなどのバイナリファイル（それ以外のファイルも可能ですが）の変更点を差分として抽出し、旧版のファイルにその差分を適用することによってアップデートされた新版のファイルを作り上げる、バイナリ差分抽出／アップデートプログラム**Bdif/Bup**(ひがしでさん作)があります。テキストファイルの差分抽出／アップデートプログラムとしては、UNIX上でよく使用される**diff/patch**(FSF作／MadPlayerさん他、多くの方が移植)も移植されています。

## 1-5-2 各種ツール

### (1) ファイラ

日常パソコンを使用していると、ファイルに対するコピー・移動・削除、あるいはテキストファイルの内容確認などのファイル操作を行うことが頻繁にあるかと思います。これらの操作はHuman68kのコマンドである、COPY、MOVE、DEL、TYPEなどを用いることによっても実現できますが、MS-DOSをまねたこれらのコマンドは、お世辞にも操作性がいいものとは言えません。この操作性に対する改善策の1つとして、SX-WINDOWなどのウィンドウ環境を用いることが考えられます。しかし、ウィンドウ環境は「メモリを消費する」「ウィンドウ画面の切り替えに時間がかかる」「マウスの操作が面倒である(キーボードのほうが慣れると速い)」などの理由から敬遠する人もいます。そこで、もう1つの改善策として考えられるのが、ファイラあるいはファイルセレクタなどと呼ばれる、ファイル操作用のユーティリティを用いる方法です。

このファイラとして最も有名なフリーソフトウェアは、PC-9800シリーズ他のマシン用に開発された、FDというソフトウェアでしょう。X68000用にも、**Fu**(☆かずりん☆さん作)や**FDX**(ヒイナさん作)のように、このFDの操作性を手本にしたファイラが存在します。他機種でFDを使用している場合も、違和感なく使えることと思います。

また、もう1つ特徴的なファイラとして、つねに2つのディレクトリの内容を表示・確認しながらファイル操作ができる「2画面ファイラ」と呼ばれるものがあります。これのおそらく元祖であり、最も有名なものとして、**TF**(Thuleさん／ippohさん作)があります。類似の発想・操作性を持つファイラとして、**MF**(雅さん作)、**WS**(Fuchiさん作)などもあります。Fuも、最近の版では2画面モードをサポートするようになりましたし、やは

り2つのディレクトリを同時に見ながらファイル操作ができるのはなにかと便利なようです。さらにDI(OOYAMAさん作)のように、高機能で、ファンの多いファイラもあります。

### (2) ファイルビューワ

テキストファイルの内容を見るのに、Human68kではTYPEというコマンドを用意しています。しかし、1画面以上の量がある場合は当然画面外にスクロールして読めなくなってしまいます。これに対処するためには、MOREなどのフィルタを用いる方法もありますが、画面をさかのぼって見ることはできません。EDなどのエディタでファイルを読み込んでもいいのですが、内容を見るだけの場合はうっかり書き換えてしまう可能性もあり、今ひとつです。そこで、ファイルの内容を確認するためのファイルビューワと呼ばれるプログラムがあります。

ファイルビューワとして広く知られたものに、UNIX上で使用されるlessというコマンドがあります。これは同じUNIXのmore(Human68kのMOREは、これの機能縮小版のようなプログラムです) を、画面の上方向にもさかのぼっていけるように拡張したものです。X68000にも多くの方(Natchさん他)が移植されています。

また、X68000の特殊機能を利用したオリジナルのファイラも存在します。view(ずんさん作)は、目にも止まらぬスピードで画面スクロールを行います。X68000はテキスト表示用のVRAMがビットマップであるため、画面出力の処理が重く、低速になりがちですが、viewはそんな常識を見事に覆してくれます。FI(OOYAMAさん作) は、きれいで見やすいスムーズスクロールをうまく使ったビューワです。あと、前述のファイラにも、内蔵のファイルビューワを備えたものがあります。ファイラから使う場合は起動が速くて便利でしょう。

また、ちょっと変わったビューワ(?)として、LHAで圧縮されたアーカイブファイルの中のテキストを、展開することなく、そのまま読めるプログラムもあります。SEE(HARPOONさん作)やlhv(けんぢさん作)などがその代表です。これらのプログラムで大きなアーカイブファイルや複数のドキュメントをまとめたものなどを見ることによって、ディスクスペースと展開する時間を節約することができます。

### (3) エディタ

X68000は、添付のOSであるHuman68kに最初からフルスクリーンのエディタであるEDが装備されています。これは発売開始当時は大変恵まれたエディット環境といえました。今でこそ、OSのマシンへの添付やフルスクリーンエディタのOSへの付属は珍しいことではなくなりましたが、現在でもかなりのマシンの場合はOSは別売であり、またOSに付属するエディタは、長い間ラインエディタという場合が一般的でした。

EDは、エディタの定番であるWordMasterに類似した操作性と、簡単なテキストの編集をするには十分な機能を持った（少なくともマシンの付属品としては）評価の高いエディタですが、いかんせん、画面のスクロールなど、直接操作性にかかわる部分の処理スピードが十分でないため、長時間の操作や大きなファイルの編集にはストレスが溜まることもありました。そこで、これらの問題点を解消するため、多くの方がEDの改造・拡張を行っています。

たとえば、**SuperED**(T.Nishikawaさん作)などです。これらのエディタは、EDと(上位）互換性のある操作性を保ったまま、スクロールなどの高速化と、さまざまな機能拡張が行われており、EDの操作に慣れている多くのユーザに愛用されています。

また、フリーソフトウェアのエディタとして有名な**MicroEMACS**(D.M.Lawrenceさん他作／likaさん他移植)もX68000に移植されています。MicroEMACSは、UNIXマシン用として有名なエディタであるEmacsを手本としてパソコン用に開発されたエディタです。分割画面による複数ファイルの編集など、Emacsの持つ便利な特徴を取り込みつつ、資源の少ないパソコンでも動作するようになっています。オリジナルの英語版にも複数の作者が存在するようですが、X68000にもかなり早い時期からさまざまな方の手で移植・改良がなされています。最近の版である、MicroEMACS Ver.3.10J1.43は、本家Emacsからdired、LaTeX、shellなどの各種モードを取り込む一方、X68000の機能をフルに活用した高速スクロールや、複数の画面モードに対応するなど、他機種版とは一味も二味も違った素晴らしい仕上がりになっています（ただ、そのかわりといってはなんですが、すでにプログラムサイズ的にはMicroの名前が当てはまらなくなってきていますが)。

Emacsを手本としたエディタにはMicroEMACSの他にもいくつかあり、X68000には**Ng**(B.Larsonさん他作／澤柳さん他移植)が移植されています。こちらは本家のEmacs(Nemacs)により近い操作性を実現しています。また、同じくUNIXのエディタであるviのパソコン版である**stevie**(T.Andrewsさん作／GAPOさん他移植)や**elvis**(S.Kirkendallさん作／GAPOさん他移植)などもX68000用に移植されています。

パソコン用の簡略版では物足りない、という方には、**Nemacs**（日本語GNU Emacs)そのものもX68000には移植されています(R.M.Stallmanさん作／icamさん移植)。最近では386クラスのCPUとDOS-Extenderなどの普及により、MS-DOSの世界でもUNIXからの移植は盛んに行われていて、NemacsもDemacs（DOS版GNU Emacs）として移植されています。X68000はリニアなアドレス空間を持っているCPUの恩恵で、UNIXからの移植は以前から行われており、Nemacsはその最たるものともいえましょう。ある程度の知識とメモリとハードディスクの残り容量があれば、「なんでもできるエディタ」「エディタの顔をしたLisp言語」などといわれるEmacsの世界をあなたも体験できるわけです。

他にも、行端での折り返しができる**edt**(いろもの物理学者さん作)など、素晴らしいエ

ディタはいろいろあります。エディタは直接マンマシンインタフェースにかかわる部分なので特徴的なものが多く、その選択は各人の好みが大きく反映されます。その意味で選択の幅は広いほうがよいのですから、さまざまなエディタがフリーソフトウェアとして手に入ることはありがたいことだといえましょう。

### (4) ディスクユーティリティ

フロッピーディスクなどを取り扱うためのユーティリティソフトも各種あります。まず、Human68kのDISKCOPYコマンドのようにフロッピーディスクを複写するためのツール、すなわちディスクコピーツールが挙げられます。たとえば、**DC**(Pop.さん作)、**hcpx**(ICHIさん作)、**xcopy**(T.Nishikawaさん作)、**acopy**(みるくさん作)などです。これらのツールの特徴は、フロッピーディスクの複写が高速である、メモリが十分にある場合はコピー元のディスクの内容を一度にメモリ上に読み込んで、何枚ものディスクに繰り返し書き出す、フォーマットを行いながら複写する、セクタをスライドさせてアクセスの速いディスクにして複写する、2ドライブに交互に複写する、Human68k以外のフォーマット（たとえばOS-9など）のディスクも複写できるなど、ツールによってさまざまです。

また、ディスクの内容を直接変更するようなツールもいろいろあります。たとえば、**dedit**(Extさん作)は、ディスクの内容をセクタ単位で変更したり、特殊フォーマットのディスクを修正したり、間違って削除してしまったファイルの復活を支援したり、とディスクエディタの名前が示すとおりのさまざまな機能を提供しています。

あるいは**refreshg**(清水さん／GORRYさん作)のようにディスク上のファイルを並べ替えてファイルアクセスを高速化するツールもあります。

また、**undel**(おおすずさん作)のように、削除ファイルの復活を自動で試みるツールや、**fsck**(Extさん作)のようにファイルシステムの異常を検出・修正するツール、あるいは**tsort**(SHUNAさん作)のようにディレクトリのエントリを直接ソートして書き換えてしまうツールもあります。いずれも通常のツールでは実現できないことばかりで、いざというとき（たとえば、ディスクが壊れたような場合など）にひじょうに重宝します。

### (5) 圧縮ツール

「1-5-1　通信関連」で述べたLHAの他にも、ディスク容量の節約などの目的で使用される圧縮ツールはいろいろあります。たとえば、**LZX**(F&Iさん作)は実行ファイルを実行可能な状態のままで圧縮します(MS-DOSでいうところの、LZEXEに相当するプログラムです)。これによって、容量の少ないディスクにも多くの実行ファイルを入れることができるようになります。あるいは圧縮をしながらバックアップを行う**asyukun**(ともちゃん作)などもあります。

## 1-5-3 常駐プログラム・デバイスドライバ

一度実行するとメモリ上に残り、OSの機能を拡張して高速化したり、操作性を向上させたり（あるいはもっと別の動作をしたり）するようなプログラムを「常駐プログラム」（TSR：Terminate but Stay Resident）と呼びます。また、システムの起動時にCONFIG.SYSの記述に従って組み込まれ、やはりシステムの拡張を行うデバイスドライバと呼ばれるプログラムもあります。これらは基本的にMS-DOSのものと同様と思ってもらってかまいませんが、メモリ空間がリニアなX68000では実メモリに余裕がありさえすれば、いくらでもこれらのプログラムをメインメモリに置いたままにしておけるため、これらによる機能拡張がひじょうに盛んです。標準のシステムにもFLOAT2.XやPCMDRV.SYS等(ちなみに、前者は、常駐プログラムとしても、デバイスドライバとしても使用できます。最近は、この形式のプログラムが多いようです）のプログラムが付属していますが、フリーソフトウェアはこれらのプログラムの宝庫です。

### (1) 画面表示高速化ドライバ（コンソールドライバ）

X68000はテキスト画面がビットマップ形式をとっているため、フォント変更などの自由度が高い半面、スクロール等の、広い画面に対する処理が重いという弱点があります。特に初期のHuman68k ver.1.xxでは画面のスクロール等がひじょうに遅く、大きな問題となっていました。

これに対処するため、「コンソールドライバ」などと呼ばれる画面処理高速化プログラムがフリーソフトウェアとして開発されました。代表的なものとして**tc**(Turbo Console、みるくさん作)や**condrv**(卑弥呼☆さん作)、**hst**(シュガーさん作)などが挙げられます。これらは、単にコンソールをひたすら高速化したものから、コンソールの出力を常時バッファリングし、あとで容易に見ることができるようにする機能（MS-DOSマシンのXSCRIPTのような機能）を追加したものなど、多岐にわたります。Human68kがver.2.xxとなり、IOCS.Xが添付されてコンソールの大幅な高速化が行われたため、以前より必然性は減ってはいるものの、さらなる高速化やバッファ機能などを備え、やはりX68000にはなくてはならないフリーソフトウェアといえましょう。

コンソールドライバとは若干趣きが異なりますが、IOCS.Xをもとにコンソール関係を含むIOCSのさらなる高速化と使い勝手の向上を目指して、**HIOCS**(YuNKさん作)というプログラムも開発されています。これは、常駐プログラムではつねに問題となる、他のプログラムとの相性も標準のIOCS.Xより向上している等、ひじょうにお勧めのソフトです。

### (2) 浮動小数点演算ドライバ

X68000は、浮動小数点演算をドライバとしてプログラムから独立させ、これを入れ替えることによって数値演算プロセッサ68881に対応したりしています。この方法は、コプロセッサの対応をプログラム側で考える必要がない、などの利点があるのですが、半面、ドライバの性能によってプログラムのスピードが左右されるという問題があります。特にHuman68k ver.1.xx付属の浮動小数点演算ドライバであるFLOAT2.Xは、かなり処理速度の遅いドライバで、そのため、高速化を施したさまざまなドライバが作成されました。たとえば、**float2p**(Pop.さん作)などです。浮動小数点演算ドライバをこれらのものに入れ替えるだけで動作が見違えるほど速くなったものです。FLOAT2.Xもver.2となって大幅に高速化したため、以前ほどの差は見られませんが、やはり少しでも速いほうがうれしいものです。また、その他にもコプロセッサ68882に対応したドライバ**float50**(LUCASさん作)などもあります。

### (3) RAMディスク／キャッシュ

Human68kにはメモリを高速なディスクとして使用するRAMディスクドライバとして、標準でRAMDISK.SYSが付属しています。しかし、このドライバはRAMディスクとしてはそれほど高速でないこともあり、さまざまな改良をほどこしたRAMディスクドライバがフリーソフトウェアとして出回っています。たとえば、**GRAD**(GORRYさん作)、**FLEXDISK**(T.Nishikawaさん作)、**tmd**(みるくさん作)などです。機能的にも、容量の動的変更や切り離しなど、さまざまな機能をサポートしています。

また、一度読み込んだディスクの内容をメモリ上に保存して再アクセスのときに使う「ディスクキャッシュ」と呼ばれるソフトもいろいろ出回っています。SCSIディスクにも対応している**DCACHE2**(Arimacさん作)がお勧めでしょう。

### (4) キーボードの割り付け変更

X68000のキーボードは、[かな]キーや[CAPS]キーなどの特殊キーがひじょうに不便な位置にあり、大変使いづらいものとなっています。そのため、これらのキーを入れ替えるような常駐プログラムも多く存在していますし、日本語FEPであるFIXERにもそのような機能が搭載されていたりします。さらに進んで、ほとんどのキーの位置を自由に再配置できるような、**keymap**(卑弥呼☆さん作)や**CAPS**(ながいさん作)などのプログラムも存在します。これらを用いれば、他のマシンのキーボードと同じ配置にしたり、全然異なった並びのアルファベット配列にしたりすることも可能です。

### (5) システムの機能拡張

常駐プログラムの中には、OSであるHuman68kの制限を解除したり、機能を拡張した

りするプログラムも数多く存在します。これらのプログラムが常駐した場合には標準のシステムと完全に互換とはいえなくなり、たとえば、そのプログラムが常駐していないシステムでは正しく読み出すことができないディスクを作成する、などの可能性があります。そのため、取り扱いには注意が必要ですが、やはり、一度使いはじめると、その便利さから手放せなくなってしまいます。

たとえば、NNPCMDRV(GORRYさん作)、MOPMDRV(MEW.さん作)、history2(TONBEさん作)などのように、標準添付のドライバを改良した、あるいは上位互換のものを作ったドライバが多数存在します。これらのソフトは、今使っているドライバと入れ替えるだけで使えるのでとても便利です。

また、この種の機能拡張プログラムとして挙げられるものに、TwentyOne(Extさん作)があります。Human68kのファイルシステムはMS-DOSとほぼ互換性があり、そのため、国内のMS-DOSマシンで書き込まれた5インチ2HDの標準フォーマットフロッピーは、まず問題なく読み込めます。そのうえでHuman68kではファイル名を21文字まで使える、小文字もファイル名に使用できる、などの拡張がなされています(そのため、Human68kで作成したフロッピーはそのままではMS-DOSマシンでは読めないこともあります)。しかし、やはりMS-DOSとの互換性に気を遣ってか、21文字まで使えるファイル名も、8文字＋拡張子3文字までしか識別されない、大文字と小文字は別の文字として扱われ認識されない、などのひじょうに中途半端な仕様となっています。そこで、TwentyOneはメモリ上のHUMAN.SYSにパッチを当てることによって、これらの制限を取り払ってしまおう、というプログラムです。TwentyOneを用いることによって、上記の制限解除をはじめ、“＝”などのファイル名に使用できないキャラクタを使用可能にしたり、複数のピリオドをファイル名に使用できたり、と大幅な制限緩和が可能になります。これによって長いファイル名を使っても、これまでのように途中までしか認識されず、他のファイルと混同される心配もなく、自由に使えるようになります。

また他にも、シンボリックリンク（UNIXの機能の1つで、他のドライブやディレクトリにあるファイルを、あたかもカレントディレクトリにあるかのように利用できる機能のこと）を実現するドライバであるlndrv(沖　勝さん作)やlink(小笠原博之さん作)とか、x、r、z以外の拡張子のファイルも実行可能にするドライバexecd(沖　勝さん作)など、さまざまなものがあります。

### (6) SRAM常駐ソフト

X68000には、小容量ながらバッテリバックアップされたSRAMが搭載されています。このSRAMを活用するためのプログラムとして、標準ではSRAMDISK.SYSのRAMディスクドライバが付属するのみですが、フリーソフトウェアでは、このSRAMに常駐するものがいろいろあります。たとえば、通常の常駐ソフトとは若干意味合いが異なりますが、

起動デバイスをメニューで選択できる**BM**(FOX&MIXさん作)などもSRAMを活用するフリーソフトウェアです。ハードディスクで複数のパーティションから起動するような場合に便利です。

**(7) その他の常駐ソフト、関連ツール**

この他にもX68000にはさまざまな常駐ソフト・デバイスドライバがあります。たとえば、特殊フォーマットされた容量の増えたフロッピーディスクを使う**9scdrv**(6no8rouさん作)、**2HD**(Missy.Mさん作)などのフロッピーディスクドライバや、バックグラウンドでプログラムを走らせるためのマルチタスク用ドライバ**bgdrv**(KeIさん作)、ファイル名の入力をサポートする**tfs**(卑弥呼☆さん作)、**fiss**(annalisaさん作)などの常駐セレクタ等、枚挙にいとまがありません。また、コマンドラインからのデバイスドライバの組み込み・切り離しを行う**ADDDRV**(去石さん／山本さん作)など、関連ツールもいろいろ揃っています。

## 1-5-4 シェル

OSに対して人間が直接コマンドを入力して操作を行うためのプログラムを「シェル」といいます。Human68kではCOMMAND.Xがこれにあたります。このシェルもフリーソフトウェアとしてさまざまなものが出回っています。**minsh**(Mad Playerさん作)や**fish**(板垣さん作)など、UNIXのシェルをモデルにしたものが多いようです。また、付属のCOMMAND.Xにパッチを当てて、TwentyOne対応などの改良を行った**hcommand.x**(K-rasさん作)もあります。いずれも標準のシェルに飽きたりない方は使ってみるとよいでしょう。

## 1-5-5 開発環境

X68000の開発環境としては、シャープ純正のアセンブラとCがありますが、フリーソフトウェアにも開発用の優れたプログラムがいくつも存在します。まず、Cコンパイラとして、GNUのCコンパイラを移植し、X68000向けに独自の拡張を行った**GCC**(FSF作／まりこさん移植)があります。純正のXCよりもずっと高速なプログラムを出力する優れたコンパイラのため、事実上X68000の標準コンパイラとなっているといえます。また、アセンブラやリンカについても、純正のAS、LKと上位互換で、より高速な**HAS**(YuNKさん作)と**HLK**(SALTさん作)があり、こちらも完全に純正品にとってかわっています。特にHLKの高速さは驚異的で、はじめて使われる方は驚くこと請け合いです。デバッガとしては、やはりGNUからの移植であるソースコードデバッガ**GDB**(FSF作／momoさん移

植)があります。また、プログラムの解析用のツールとして、アセンブラのソースコードジェネレータ**dis**(Abechanさん作)があり、驚異のソースコード再現率を誇っています。

## 1-5-6 グラフィック・音楽関連

X68000の得意分野の1つであるグラフィック関連については、市販ソフトにもいろいろ優れたツールがありますが、フリーソフトウェアも負けていません。

まず、グラフィックのセーバ/ローダですが、65536色モードでは高速・高圧縮率の**PIC**(柳沢さん作)があります。このモードでは、ほぼ完全にX68000の標準フォーマットとして認められており、最近では市販のグラフィックツールも、このPICフォーマットをサポートするようになってきています。また16色モードでは、PC-9801などと互換性のあるフォーマットの**MAG**(W.Rinnさん作)が広く用いられています。

グラフィックツールとしては、65536色モードの**mfged**(結城さん作)や16色モードの**PST**(Kennaさん/PUNAさん作)などの優れたソフトが存在します。

X68000のもう1つの特徴的な分野、音楽関連でも各種の優れたプログラムがあります。音源ドライバとしては内蔵のFM音源とADPCM音源を活用した**mxdrv**(みるくさん他作)や他機種用のデータも演奏できる外部MIDI音源用の**RCシステム**(HARPOONさん/TURBOさん作)、両方に対応する**Z-MUSIC**(西川さん作)など、他にもさまざまなものがあります。また、入力用のツールとしても、ステップエディタ**STed**(TURBOさん作)などがあります。

## 1-5-7 GUI関連

X68000には標準のウィンドウシステムとしてSX-WINDOWがあります。市販ソフトもそろそろ出始めてきたようですが、このウィンドウシステム上のアプリケーションもフリーソフトウェアとしていろいろ出回っています。また、ウィンドウシステム自体をフリーソフトウェアとして開発してしまった**Ko-WINDOW**(小林さん作/小笠原さん改良)というシステムもあります。SX-WINDOWがMacintoshをかなり意識したシステムなのに対し、こちらはUNIXのX-Windowのような雰囲気を持っています。また、Ko-WINDOWベースのフリーソフトウェアも数多く発表されています。

## 1-5-8 その他

他にもここでは述べられなかったさまざまなフリーソフトウェアがあります。特にフリーソフトウェアならではのゲームや一発ギャグなど、実際に見てみるしか説明のしようがないソフトもいろいろあります。

# 1-6 フリーソフトウェアの使用・入手

## 1-6-1 付録ディスクの使い方

本書に付録として添付されているフロッピーディスクには、以下のようなフリーソフトウェアが入っています。

| | | | | |
|---|---|---|---|---|
| MUTERM | LZH | 63494 | 92-03-03 | 12：00：00 |
| TMNX | LZH | 120412 | 93-02-24 | 12：00：00 |
| TMNBASE | LZH | 204108 | 93-03-08 | 13：29：54 |
| ish121 | Lzh | 43868 | 91-04-21 | 2：04：50 |
| LHA_X633 | x | 45826 | 92-05-31 | 2：06：00 |
| BdifSet | Lzh | 60317 | 92-12-29 | 0：00：00 |
| Fu204f3_A | Lzh | 78494 | 93-02-17 | 12：00：00 |
| MF_X_206a | Lzh | 93595 | 93-01-23 | 12：00：00 |
| lzx0434ld | lzh | 46156 | 92-10-05 | 0：00：00 |
| see031_1 | lzh | 32494 | 91-07-31 | 14：18：58 |
| DC2_065 | LZH | 26408 | 92-10-21 | 0：00：00 |
| 118SUPERED | LZH | 57756 | 92-08-16 | 12：00：00 |
| tsort255 | Lzh | 19015 | 92-02-14 | 4：49：26 |
| dedt223a | Lzh | 62652 | 92-09-06 | 19：46：44 |
| 18SRAMCL | LZH | 6676 | 92-06-07 | 13：55：20 |
| TwOn123 | Lzh | 51197 | 93-01-09 | 22：59：12 |
| hcmd_05 | Lzh | 10390 | 92-11-06 | 9：30：46 |
| caps061g | Lzh | 29056 | 92-11-23 | 0：00：00 |
| FLT23_20 | LZH | 39930 | 92-09-03 | 1：08：24 |
| HIOCS15 | LZH | 22701 | 93-02-15 | 0：55：28 |
| 103FLEXDISK | LZH | 11230 | 92-03-27 | 12：00：00 |
| dcache | lzh | 31532 | 92-09-19 | 0：48：16 |
| de0015 | lzh | 11929 | 92-11-02 | 17：17：08 |
| drv_210 | lzh | 31768 | 89-10-20 | 2：10：10 |
| drv211fx | lzh | 4399 | 89-11-06 | 15：14：30 |
| CDINIT | LZH | 18397 | 91-12-14 | 0：01：10 |

これらのソフトウェアを使う前に、まず、しておかなければならないことがあります。それは、

**ディスクのバックアップをとっておく**

ということです。フロッピーディスクは物理的・電磁気的な衝撃にひじょうに弱いメディアです。ご使用前に必ずバックアップをとり、以後はそちらを使って作業をするようにしてください。オリジナルディスクは、バックアップディスクが壊れた場合のためにプロテクトシールを貼って大切に保管しておいてください。バックアップはHuman68kのFORMATコマンドでフォーマットした空きディスクに、DISKCOPYコマンドでオリジナルディスクを複写することで作成できます。詳しくは Human68kのユーザーズマニュアルを参照してください。

バックアップを作成したら、いよいよ作業開始です。

まず、ディスクの内容を展開するための作業用空きディスクを準備します。一番いいのはRAMディスク上で作業を行うことですが、プログラムによっては、ある程度(数百Kバイト）の大きさが必要になりますので、メモリに余裕のない方はハードディスクなり、フロッピーディスクなりで作業を行ってください。フロッピーの場合は展開にかなり時間がかかるかもしれません。

以下の説明では、本書添付のフロッピーディスクの入っているドライブをB:ドライブ、作業用の空きディスクをC:ドライブとして説明を進めていきます。ドライブ名が異なる場合は各自のシステムにあわせて読み替えてください。空きディスク上での作業は、ルートディレクトリで行うものとして説明しますが、ハードディスクを使っている場合等、ルートディレクトリで作業をするのは避けたいときには作業用のディレクトリを作成し、その中で展開作業を行ってもかまいません。また、作業の説明はCOMMAND.Xのコマンドライン上で作業するものとして行いますので、SX-WINDOWをお使いの方はCOMMAND.Xを起動して作業してください。

**①LHA.xのインストール**

本書添付のフロッピーディスクのプログラムは、すべてLHAというプログラムで圧縮してあります(各ファイルのLZHという拡張子は、アーカイブされたファイルであることを表しています）ので、使用する前に通常のファイルに展開しなければなりません。そのためには、まずLHAが必要になります。そこで、本ディスクの中で唯一、拡張子がxであるファイル、

```
LHA_X633    x    45826  92-05-31   2:06:00
```

を作業用ディスクにコピーしてください。具体的には、

```
C> COPY B:LHA_X633.x C: [CR]
B:LHA_X633.x
      1 個のファイルをコピーしました
```

とします（以下、本書では、キー入力部分をアンダーラインで表し、リターンキーを押すことを[CR]と表します)。このファイルは、LHA自身を自己解凍型という形式のファイルに圧縮したもので、 LHAがなくても展開できるようになっています。展開の方法は、プログラムを実行すればよいだけです。具体的には、このファイルを実行すると、以下のようなメッセージを出力して自動的に展開が行われます。

```
C> LHA_X633 [CR]
LHA's SFX 0.05 OKADA, Norio

LHA.doc ・
テク.LHA ・
ReadMe.LHA ・
LHA.x ・・
LHA.bfd ・
```

展開が終了すると、作業ディスク上には以下のようなファイルが新たに作成されているはずです。DIRコマンドを用いて確認してください。

```
LHA      doc    13772  92-05-31  2:06:00
テク     LHA     3721  92-05-31  2:06:00
ReadMe   LHA     2269  92-05-31  2:06:00
LHA      x      60000  92-05-31  2:06:00
LHA      bfd      977  92-05-31  2:06:00
```

この実行ファイル、LHA.xが展開作業に必要なファイルです。その他のファイルはLHAのドキュメントなどです。TYPEコマンドで表示するなり、プリントアウトするなりして読んでください。とりあえず実行ファイルLHA.xは、ふだん使っているHuman68kのコマンドが入っている、実行パスの通ったディレクトリにコピーしておいてください。RAMディスクで作業をしている場合などは、このままにしておくと電源オフでファイルが消えてしまうからです。

たとえば、他のコマンドが“A:¥BIN”のディレクトリにある場合は、

```
C> COPY C:LHA.x A:¥BIN [CR]
C:LHA.x
        1 個のファイルをコピーしました
```

という具合です。これで他のHuman68kのコマンドと同様に、コマンドラインからタイプ

するだけでLHAを実行できます。LHAは、コマンド名のみを実行すると、ヘルプメッセージを出力するようになっています。試しに作業用ディスクの内容を整理してから実行してみましょう。

```
C> DEL LHA*.* [CR]
C:¥LHA*.*を削除します
 よろしいですか <Y/N> Y
LHA_X633.x
LHA.doc
LHA.x
LHA.bfd
C> DEL *.LHA [CR]
C:¥*.LHAを削除します
 よろしいですか <Y/N> Y
テク.LHA
ReadMe.LHA
C> LHA [CR]
```

これで以下のメッセージが出力されるはずです。

```
  ＬＨＡ Version 2.13.00 ( MS-DOS 版) Copyright(c) H.Yoshizaki (吉崎 栄泰), 1988-92
         Version 2.06.01 (Human68k版) Special 版 by ともちゃん (岡田 紀雄), 1990-92

==== <<< 高圧縮書庫管理プログラム >>> =========================== 1992-05-31 ========

書  式：LHA [aufmdexplvst] [-rwxmpcnitzoha[-+0123lstr]] FileName.Lzh [dir¥] [Files]

-------------------------------------------------------------------------------------
《 命   令 ; `-' を頭に付けない 》 ( default = l; set LHA_CMD )
    a : 書庫にファイルを追加           u : 書庫にファイルを追加   ( 日時照合付 )
    f : 書庫のファイルを更新           m : 書庫にファイルを移動   ( 日時照合付 )
    d : 書庫内のファイルの削除         p : 書庫内のファイルの閲覧  ( set PAGER )
    e : 書庫からファイルを復元         x :  e に同じ        ( ディレクトリつき )
    l : 書庫の一覧表示                 v :  l に同じ        ( ディレクトリつき )
    s : 自己解凍書庫の作成             t : 書庫内のファイルのエラーチェック
《 スイッチ ; `-' を頭に付ける 》  ( set LHA )
    r : サブディレクトリも検索         w : ワークディレクトリの指定 ( set TEMP )
    x : ディレクトリ名を有効にする     m : 問い合わせを行わない
    a : ファイル属性の保存・復元をする c : 日時照合を行わない
    n : 経過・リスト表示の種類変更     i : 大文字／小文字の区別をする
    t : 書庫の時刻を最新のファイルに   z : 無圧縮格納        ( set LHA_NON_SUF )
    o : 従来の -lh1- 圧縮形式で格納    h : ヘッダ形式の指定      ( default = 1 )
    p : 名前の比較・表示を厳密に行う   e : 空ディレクトリ非対応書庫作成

=====================================================================================
 詳しくは付属の document を読んで下さい                NIFTY-Serve     TBE01054
<<< NET への転載は自由です >>>                         ともちゃんＮＥＴ        1
```

出力されない場合はLHAが実行できる状態になっていません。実行ファイルの入ったディレクトリ（ここでは“A:¥BIN”）にLHA.xがあるか、そのディレクトリにパスが通っているか（PATHコマンドで確認できます）などをチェックしてください。

**②目的プログラムのインストール**

無事LHAが実行できる状態になったら、あとは使用してみたいプログラムを作業ディスクにコピーして展開するだけです。各圧縮ファイルの展開は、LHAのxコマンド(127ページ参照)で行います。たとえば、MuTermを展開する場合は、MUTERM.LZHを作業ディスクにコピーしてきて、LHAをxコマンドで起動します。つまり、

```
C> COPY B:MUTERM.LZH C: [CR]
B:MUTERM.LZH
        1 個のファイルをコピーしました
C> LHA x MUTERM.LZH [CR]

Extracting from Archive : MUTERM.LZH

MUTERM.X        : 解凍終了  [   51200 ] (   51200 )
MUTERM.CNF      : 解凍終了  [    2597 ] (    2597 )
MUTERM.DOC      : 解凍終了  [    7526 ] (    7526 )
MUTERM.MAN      : 解凍終了  [   60022 ] (   60022 )
GETFONT.COM     : 解凍終了  [    4028 ] (    4028 )
GETFONT.DOC     : 解凍終了  [    4065 ] (    4065 )
ESC.DOC         : 解凍終了  [    2880 ] (    2880 )
```

という要領です。展開が終了したら、必要なファイルをインストールするディレクトリにコピーしてください。LHAの詳細や、各コマンドのインストール方法などは、第2章および各圧縮ファイルに付属しているドキュメントを参照してください。LHAを実行可能にしておけば、あとのファイルの展開は一度にやらなくても、使いたいときに必要なファイルを作業ディスクで展開すればいいでしょう。

また、TMNなどのようにディレクトリごと圧縮されているものの場合は、作業ディスクで展開してからインストールするディレクトリにコピーするよりも、最初からインストール先で展開したほうがインストールが楽になるでしょう。

## 1-6-2　パソコン通信の準備

添付ディスクに収録したプログラムは、極力その時点での最新版を収録するようにしましたが、書籍の付録としてつける以上、アフターサポートや定期的なバージョンアップなどは残念ながら事実上不可能です。また、すでに書いたとおり、これらのプログラムはフリーソフトウェアという氷山のほんの一角であり、同種のプログラムやさらに別のプログラムが山のように存在します。特に今回はシステムまわりのフリーソフトウェアに収録を

絞ったこともあり、GCCなどのプログラム開発関係やグラフィック、ミュージック関係などのアプリケーション、あるいはゲームやSX-WINDOW用のプログラムなどは、残念ながらまったく収録されていませんし、本文でもほとんど触れていません。

これは本書が、主に添付するディスク容量（2HD=1.2Mバイト）の関係から「すべてのフリーソフトウェアを収録することは不可能なのだから、とりあえずパソコン通信でプログラムが入手できる環境と、1-5で紹介したようなジャンルの一部からいくつかのプログラムを収録して、興味がわいた方は自分でパソコン通信を始めて収録できなかったフリーソフトウェアを入手してもらおう」という方針で編集されているからです。

ですから、本書とその添付ディスクでフリーソフトウェアやパソコン通信に興味を持たれた方は、ぜひパソコン通信を始めてみましょう。ここでは、そのために必要な準備などを説明します。

### (1) 用意するもの

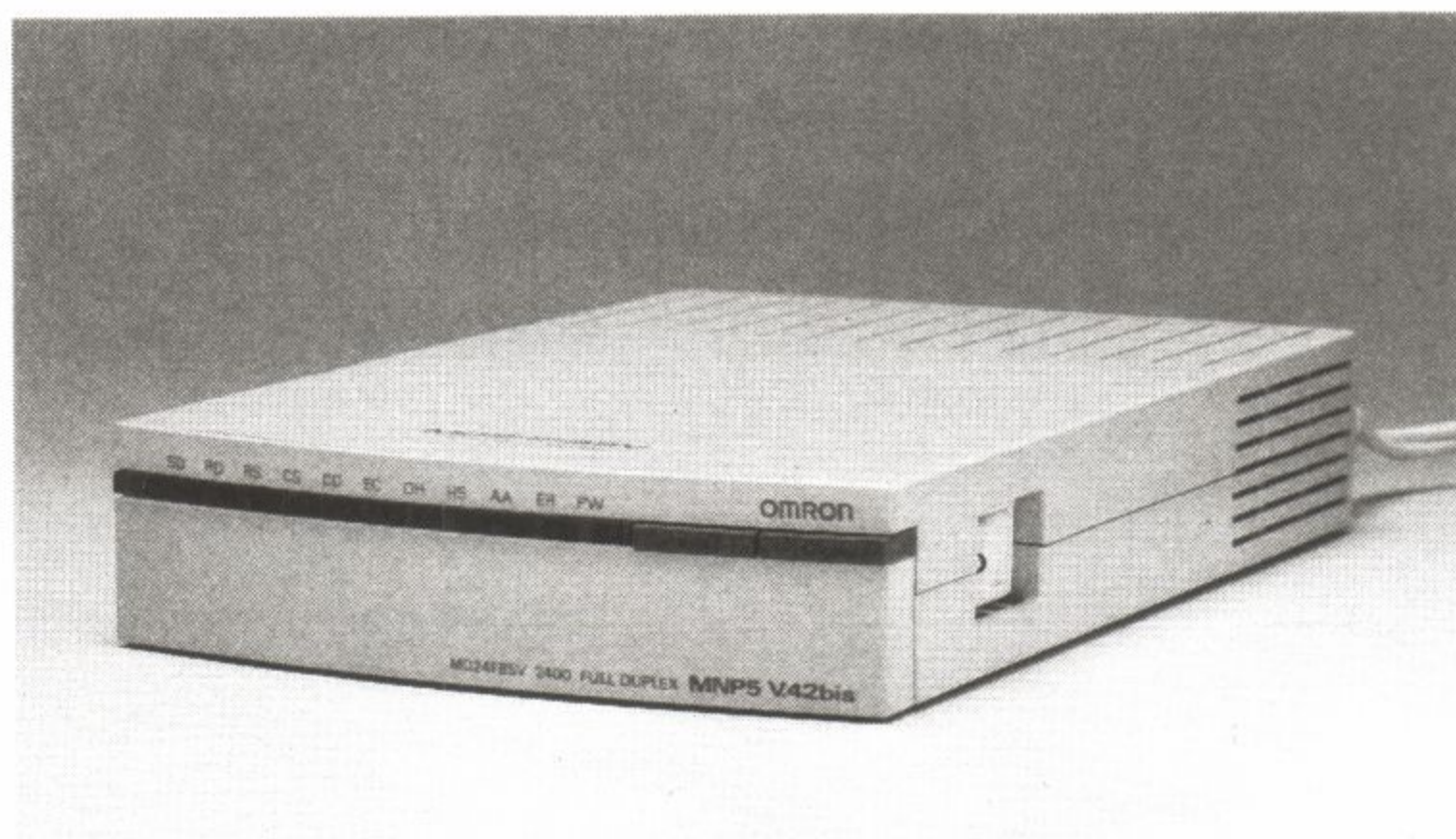
X68000本体やディスプレイなどはすでにあるものとして、パソコン通信を始めるために新たに必要なもの（ハードウェア）には以下のようなものがあります。

#### ・モデム

パソコン通信は電話回線を使って行われるため、パソコンで使用できるデジタル信号を電話のアナログ音声信号に変換する（あるいは、その逆にアナログ信号をデジタル信号に変換する）ための装置です。通信速度などによってさまざまな種類があります。詳しい説明はパソコン通信の書籍やモデムのマニュアルなどを見ていただくとして、とりあえず、

- 全二重タイプのもの（パソコンショップなどで売っているモデムはこのタイプです）
- ATコマンドを受け付けるもの
- 通信速度は2400bps（CCITT V.22bis）以上のもの
- エラー訂正、データ圧縮機能としてはMNPクラス5とCCITT V.42bisをサポートしているもの

を選んでください。最近では、上記の機能を持たないモデムは一般にはほとんど市販されていません。製品にもよりますが、2400bps、MNPクラス5の機能を持ったもので、2万～4万円くらいで手に入るはずです。

Pht.1　2400bpsモデムの一例（オムロンMD24FB5V）

ちなみに、ここでいうATコマンドとは、モデムに与えるコマンド規格の一種で、各コマンドがATの文字で始まるものです。米国ヘイズ社がはじめに規格化し、搭載したもので、モデムのコマンド体系としては最も広く普及しています。bpsはbit per secondの略で、1秒あたりのデータ転送量を表しています。当然、数値の大きいほうが転送速度は速くなります。また、MNPクラス5やCCITT V.42bisは、それぞれ通信データのエラー訂正や圧縮を含む規格の名前です。どのような規格かはここでは説明しませんが、これらの規格がアクセス側とホスト側の両方のモデムでサポートされていると、正確で高速な通信が行えると思ってください。詳しくはモデムのマニュアルや関連書籍などを参照してください。なお、あとで触れるモデムの設定のところでも一部説明してあります。

通信速度については、2400bpsより遅い1200bpsなどのモデムでも使用できますが(その場合、普通、送受信時のデータのエラー訂正や圧縮機能はサポートされていません)、モデムの通信速度が遅いということは(特にプログラムをダウンロードする場合は)、そのまま通信時間の増加、すなわち電話料金の増加につながりますので、特にこれからモデムを購入される場合は高速なものにしたほうがいいでしょう。本書を執筆している時点では、一般にパソコン通信で主に使われているモデムは2400bpsですが、そろそろ9600bpsのものや、それ以上（14400bpsなど）のものが普及の兆しを見せてきています。ホスト側でも、これらに対応（通信速度は、通信のホスト側とアクセスする側のモデムの遅いほうの速度にあわせますから、ホスト側のモデムが高速でないと、いくら高速なモデムがあっても意味がないことになります）するところが増えはじめてきています。

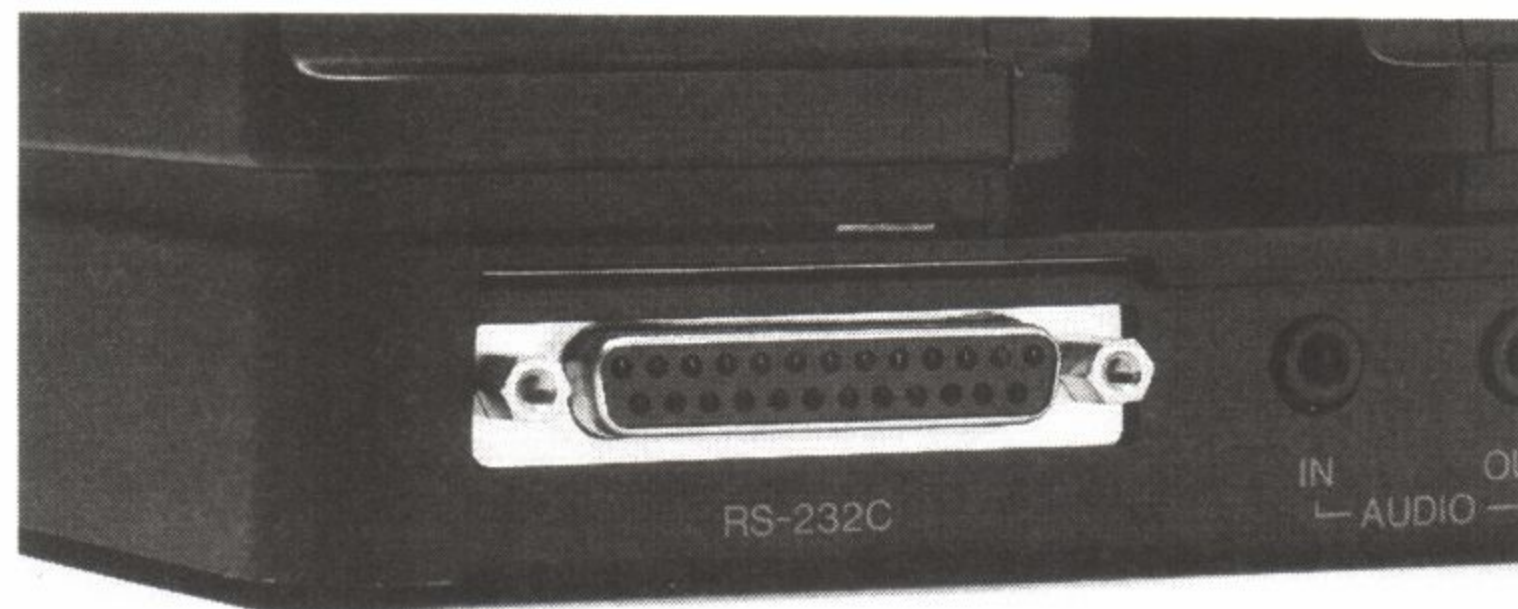
将来を見越して、これらの9600bps以上の通信速度のモデムを購入するのもいいかと思います。最近では、9600bpsモデムが5万円前後で購入できるようになり、使用者もずいぶん増えているようです。

また、最近はFAX機能などの付加機能を持ったモデムも登場してきていますが、これらを使うためにはソフトウェアによるサポートが必要です。X68000では、今のところ、それ

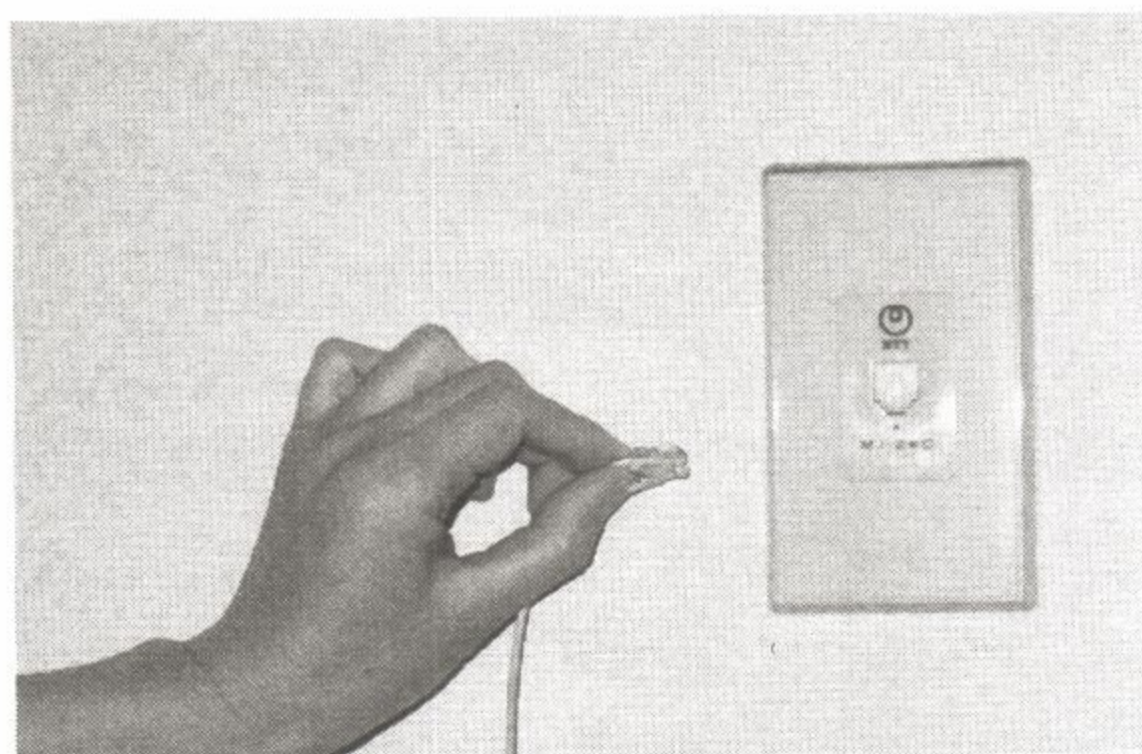
らをサポートしたソフトウェアは見当たりませんので、残念ながら、あっても活用できません(もっとも、パソコン通信になじんでくると、「ないものはそのうち誰かが作るんじゃないかな～」という安易な発想をするようにもなってしまうのですが……)。

**・RS-232Cケーブル**

モデムとX68000本体を接続するケーブルです(モデムに付属している場合もあります)。X68000側はRS-232Cのシリアルポートに接続します。注意することは、X68000側端子(D-sub25ピンメス。PC-9801などと同じタイプのもの)およびモデム側の端子に接続できるコネクタのついたケーブルであることと、内部の結線がストレート接続(つまり、ストレートケーブル)であること(クロス接続するもの、つまりクロスケーブルもパソコンショップでは売られているが、こちらはコンピュータとコンピュータを接続してデータの送受信をする場合に使う)ぐらいでしょう。モデムを買うときにお店の人に「パソコン通信をしたいのだけれど」と目的を言って、いっしょに選んでもらうと、間違いがないでしょう。

**Pht.2** X68000のシリアルポート(EXPERTの背面)

**・電話回線**

パソコン通信は電話回線を経由して行うので、当然必要です。すでに電話回線をひいてある方はそれを使えばいいのですが、新たに電話回線をひく場合はけっこうお金がかかります。通常の電話と共用できますから(もちろん、ISDN回線でない限り、電話と通信を同時に利用することはできませんが)、家に電話回線がひいてある場合は、そこからケーブルを引っ張ってくれば問題ありません。ただ、ケーブルの接続の関係でモジュラータイプ(簡単にケーブルの着脱ができるようになっているコネクタ(Pht.3参照)。これに対し、ローゼットタイプのコネクタもあり、こちらの場合は簡単にケーブルの着脱ができない)になっている必要があります。なっていない場合はNTTに連絡してモジュラータイプに切り替えるアダプタをもらってください(無料です)。

Pht.3　モジュラーコネクタ

家族と電話回線を共用する場合はあらかじめ断っておかないと、NTTからの請求書が来た際にびっくりされるかもしれません。パソコン通信はやり方にもよりますが、かなり電話代がかさむことがあるからです。通信をやっている人たちの中には毎月５ケタの電話代を払っている人も珍しくありません。パソコン通信は、自分の財布の許す範囲でやるように気をつけましょう。

なお、NTTのサービスであるキャッチホンはパソコン通信とひじょうに相性が悪いようです。キャッチホンになっていると、通信中に他からの電話がかかった場合、ホスト局との接続が切れてしまいます。電話回線自体はつながっているのですが、相手のモデムからの信号が一瞬とぎれるので、モデムは回線が切れたと判断し、通信を打ち切ってしまうからです。モデムの設定を調整することによって、ある程度対処できる場合もありますが、可能ならばパソコン通信用の電話回線ではキャッチホンの契約は解約することをお勧めします。

### (2) 前準備

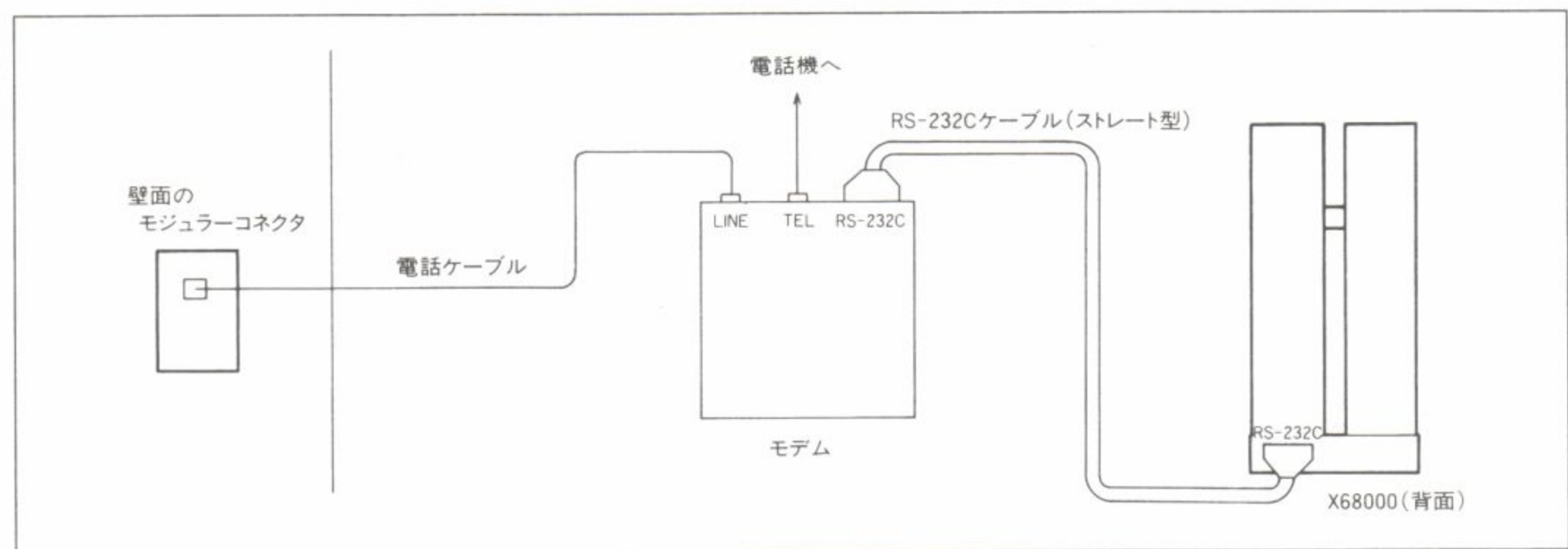
必要な器材を入手したら、今度は通信のための準備を行います。

#### ・機器の接続

まず、モデムなどの機器を通信可能な状態にする必要があります。接続方法を簡単に説明すると、X68000とモデムをRS-232Cケーブルで接続し、電話機の背面などについている電話ケーブルの接続端子（モジュラージャック。プラスチック状の接続端子になっている)を、先端についているプラスチックのツメを押さえながら引き抜き、モデムのLINE端子に差し込みます。次に、モデムに付属のモジュラーケーブルをモデムのTEL端子に差し、ケーブルの反対側についている端子を、先ほどケーブルを引き抜いた電話機の背面のコネクタに差し込みます。これで、モデムとX68000、電話機の接続は終わりです。

こうしておけば、モデムを使わないときは、これまでどおりに電話機を使うことができ

ます（最近のほとんどのモデムには通信をするときだけに電源がONになるAUTOモードがついているので、電気の無駄遣いにもなりません）。ただし、パソコン通信中に電話がかかってきても話し中になり、通話することはできません。キャッチホンの場合は通話が可能ですが、通信のほうは文字化け（パソコン通信中の画面に表示されている文字がまったく違う文字として表示されてしまい、文章として読むことができなくなる）を起こしたり、つながっている回線が切れたりしてしまいます。家族と共用する場合は、パソコン通信中は電話機が使えないことをあらかじめ断っておいたほうがいいでしょう。

**Fig.1** モデムとX68000、モジュラーコネクタとの接続



### ・ソフトウェアのインストール

モデムの接続が終わったら、次にX68000上でパソコン通信用のソフトウェア（通信ソフト）を起動します。前節でも述べたように、この通信ソフトはフリーソフトウェアの得意とするジャンルの１つなのですが、パソコン通信でそれを手に入れるためには、あらかじめ通信ソフトが必要であるという、通信を始める人には一種のジレンマが存在します（同じようなことはソフト変換用のビットコンバータishについてもいえます）。

通常は、すでにパソコン通信をやっている友人や先輩に分けてもらうか、さもなければ市販のものを買ってくるか（しかし、せっかく市販のソフトを買っても、優秀なフリーソフトウェアをパソコン通信で手に入れてしまえば、おそらく二度と使うことはないというのが筆者の経験ですが）しか方法がないわけです。実は、X68000は内部に簡単な通信ソフトを持っていてターミナルモードで立ち上げることも可能なのですが、テキストデータの送受信しかサポートされていません。つまり、フリーソフトウェアなどの実行形式になっているプログラム（バイナリデータ）の送受信はサポートされていないため、いったんバイナリデータをテキストデータに変換し、テキストデータの形でダウンロードするしか方法がありません。テキストデータでダウンロードしたプログラムを元のバイナリデータ形式に戻すには、前述したパソコン通信で手に入るishが必要であるというジレンマに陥ってしまいます。

しかし、この本を購入した方は添付ディスクに通信ソフトをはじめとする、パソコン通

信を始めるにあたって必要となるソフトウェアが一式揃っています。このディスクの中から以下のソフトウェアを使用可能な状態にしてください。圧縮されているプログラムの展開方法は「1-6-1　付録ディスクの使い方」に、各ソフトウェアのインストール方法は第2章で詳しく説明されていますので、そちらを参照してください。

(a) 通信ソフト

本書にはMuTermとTMNという2つの通信ソフトが添付されています。その特徴は先ほど簡単に触れましたし、第2章にはその解説も載っているので、好きなほうを使ってください。本節では、TMNを使用して説明します。

(b) ビットコンバータ

先ほども説明したように、バイナリ転送がサポートされていないホスト局からダウンロードしてきたプログラムを使用するためには、ishというテキスト／バイナリ変換用のプログラムが必要になります。バイナリデータの送受信がサポートがされているホスト局の場合は、サポートされていないホスト局に比べれば使用頻度は低くなりますが、いずれにしてもパソコン通信必携のプログラムといえます。本書には、ishのVer.1.21が収録されているので、これを使ってください。

(c) アーカイバ

プログラムは、大抵の場合、複数のファイルを取りまとめ、圧縮した形式でホストに登録されています。これを展開して使用可能な状態にするプログラムがLHAです。1-6-1の手順を終了した方は、すでに使用可能な状態になっているはずです。

(d) バイナリ差分アップデートプログラム

フリーソフトウェアは、ユーザの声を反映して何度もバージョンアップしながら、次第に使い勝手のいいプログラムになっていきます。このため、フリーソフトウェアの中には、バージョンアップの際、旧版の実行ファイルを新版にすべて変更した形ではなく、バイナリ差分形式（つまり、変更された部分のみ）で登録されているものもあります。このようなプログラムを使用するためには、前に登録されている旧版のプログラムと、その差分プログラムを組み合わせて（通常、「差分を当てる」などといわれている）新しい版数のプログラムにするアップデートプログラムBupが必要になってきます（もちろん、はじめてそのフリーソフトを使おうという場合には、差分ではなく、元の実行ファイルが必要ですが、パソコン通信を始めれば、すぐにBupのありがたみがわかるでしょう)。これも、本書のディスクに収録されています。詳しい説明は、第2章を参照してください。

**・モデムの設定**

通信ソフトのインストールが終わったら、今度はモデムの設定を行います。モデムの設定は、基本的に各モデムによって異なります。ここでは、比較的一般的であると思われるATコマンド、MNPクラス5を備えたモデムを例にして簡単に説明します。ただし、繰り返しますが、同じATコマンド、MNPクラス5のモデムでも、メーカによってはコマンドが拡張されていたり、同じ命令でも機能が微妙に違う場合がありますので、モデムに付属する取り扱い説明書をよく読んでください。

現在、パソコン通信で使われているモデムは、電話回線上でデータの送受信時にエラーなどが起こった場合、自動的にデータを修復したり、転送するデータをモデムが自動的に圧縮／展開を行うなど、高機能化し、その分モデムに与えるコマンドも多くなっています。

モデムの設定上注意しなければならないこととしては、以下のようなことが挙げられます。

・リザルトコードを単語形式にする
・端末速度固定モードにする
・フロー制御はパソコン－モデム間はRS/CS制御、モデム－モデム間は行わない

それぞれの詳しい内容については、モデムのマニュアル、または次のコラムを参考にしてください。

---

COLUMN

**モデムの設定**

モデムの設定の際のポイントを説明しておきます。ある程度通信に慣れたときにモデムの設定を最適にするための参考にしてください。

**(a) モデムのコマンド体系**

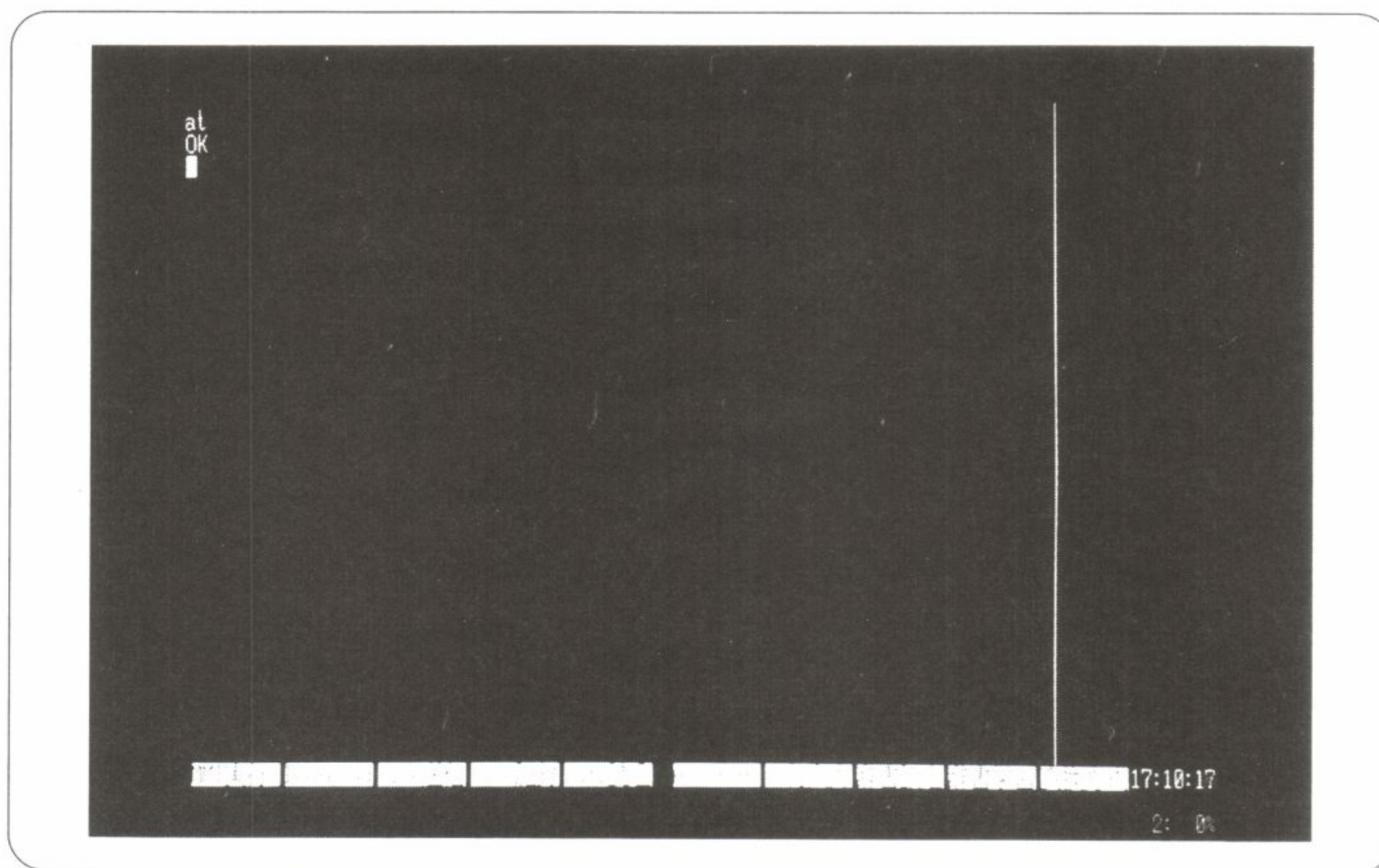
現在普及しているモデムのコマンドには２種類あります。１つはATコマンドと呼ばれるものでアメリカのヘイズ社が決めた仕様で、その後、多くのメーカが採用し、世界中で最も標準的な命令体系です。もう１つはCCITT V.25bisという規格です。CCITTは通信関係の規格標準化を提案する国際機関の名称ですが、そこで勧告されているモデムの制御命令体系がCCITT V.25bisです。国産のモデムのほとんどがATコマンドを使用できるか、もしくは両方の命令体系が使用できると思います。

TMNで使用する場合、基本的にはどちらのコマンドでもかまわないのですが、付属のMMIファイル（マクロ言語を書き込んだもの）やコンフィグファイルでは、ATコマンドのモデムを対象にして書かれています。

**(b) リザルトコード**

モデムがパソコンからのコマンドを受け取り、実行結果や現在の状態を知らせるため

にパソコンに送り返してくる文字列を、ATコマンドでは「リザルトコード」、CCITT V.25bisでは「インディケーション」と呼びます。ATコマンドの場合、単語形式か数字形式か、どちらのリザルトコードを返すかをコマンドで指定できます。TMN付属のMMIファイルは単語形式のリザルトコードを想定しています。

**Pht.4**　TMNを起動させ、ATと打ち込んでみたところ。「OK」(単語形式）というリザルトコードが返ってくる

### (c) 端末速度固定モード

MNPクラス5以上やCCITT V.42bisと呼ばれるプロトコル（パソコン通信でデータをやりとりするための手順の規格のこと。送信側と受信側でこのプロトコルがあっていないと、データの正しい送受信ができなくなる）では、送信側と受信側のモデムが自動的にデータを圧縮／展開し、総データ量を減らす形でデータ転送の速度を実質的に上げることができます。

しかし、電話回線上の実質的な転送速度が上がっても、モデムとパソコンをつなぐRS-232Cの速度が変わらないのでは、モデム間で転送されるデータの量がいくら少なくなっても無意味です。そこで、モデム間の転送速度とはかかわりなく、モデムとパソコンとの間の速度を高速で一定（一般的にはモデム間の最高速度の2倍から4倍）にしておきます。これを「端末速度固定」といいます（メーカによって名称が異なることがあります）。

端末速度固定モードになっていない場合、ホスト局とつながったときにモデム―パソコン間の速度が電話回線上の速度に変更されます。電話回線上の速度はホスト局側とパソコン側の遅いほうのモデムの速度によって決まりますので、ターミナルプログラム側の通信速度の設定も、ホスト局側のモデムの速度にあわせて変える必要があります。

TMNに付属するMMIファイルは端末速度固定で使用されることを想定しています。性能的にも端末速度固定で使用することをお勧めします。

**Fig.2**　ホスト―モデム―モデム―パソコン間の速度設定（端末速度固定の場合）

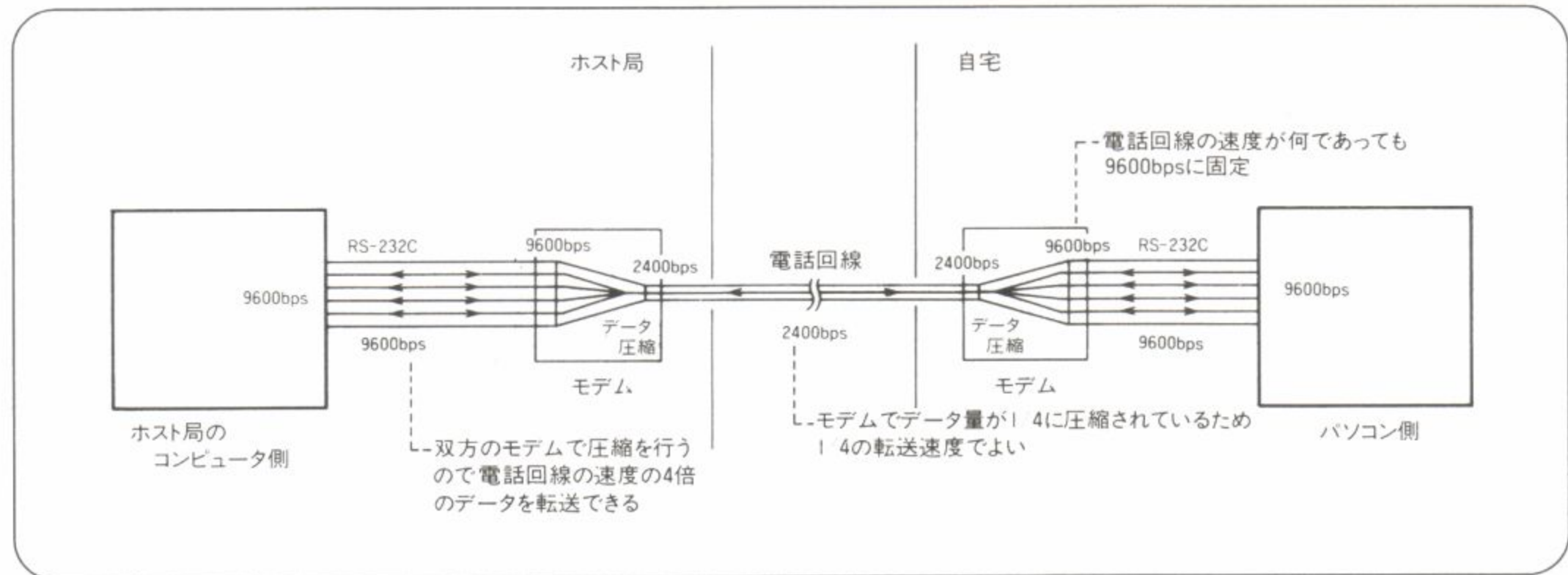
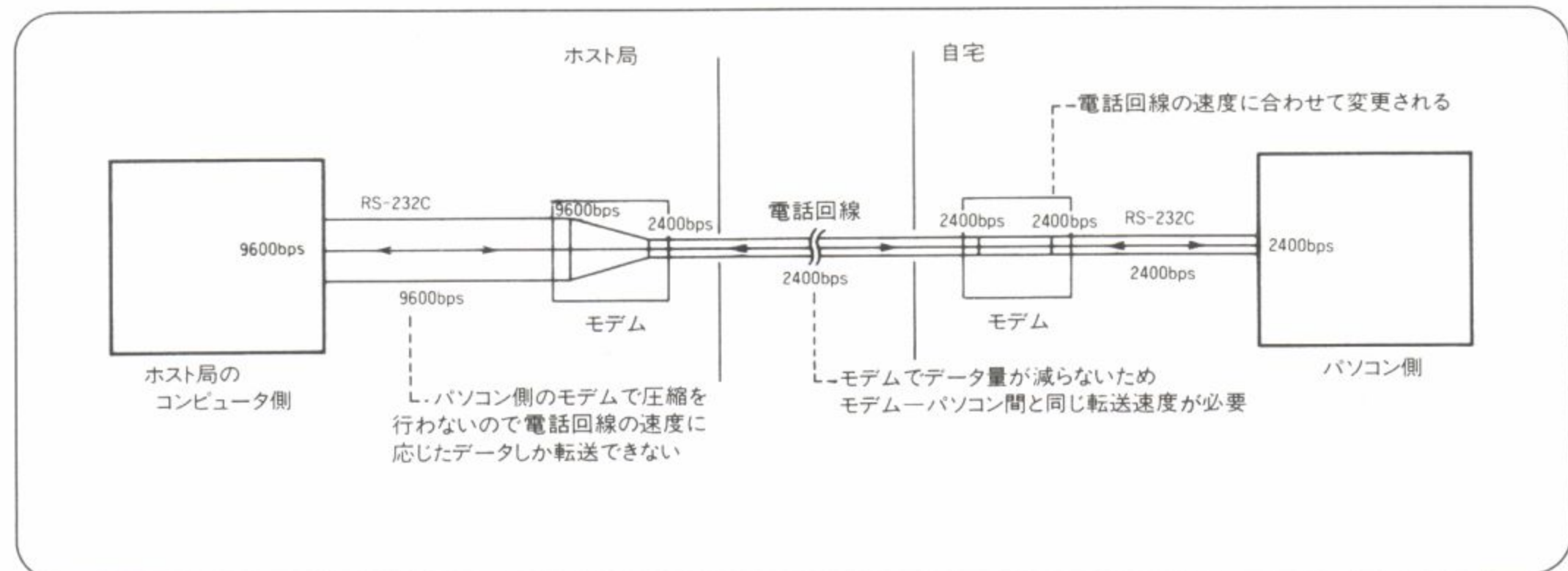



**Fig.3**　ホスト―モデム―モデム―パソコン間の速度設定（端末速度固定でない場合）

### (d) フロー制御

RS-232Cに限らず、パソコンが外部の機器とデータをやりとりする場合、受信側の処理速度が送信されるデータの速度より遅ければ、送られたデータを取りこぼしてしまいます。それを防ぐためには、受信側の処理が追い付かなくなる少し前に、送信を一時的に停止させる信号を送り、送信側は再度送信が許可されるまで待つ、という動作を行わなければなりません。これを「フロー制御」といいます。

モデムとRS-232Cを組み合わせた通信の場合、通常はXON/XOFF制御とRS/CS制御と呼ばれる2種類の方法があります。XON/XOFF制御は、送信の停止(DC3 13h)・再開(DC1 11h)を指示する文字コードを送ることでフロー制御を行います。また、RS(RTS)/CS(CTS)制御は、RS-232Cの制御用の信号ラインをハード的にON/OFFすることでフロー制御を行うため、「ハードフロー制御」とも呼ばれます。またMNPなどのモデム―モデム間プロトコルやバイナリ転送用のプロトコルも独自のフロー制御を行います。

パソコン通信の場合は、パソコン―モデム間とホスト―コンピュータ間のフロー制御が問題となります。前者はRS-232Cで接続されているのでXON/XOFF制御とRS/CS制御のどちらも使用可能ですが、後者の接続は電話回線なので、フロー制御を行う場合はXON/XOFF制御で行うことになります。

ホスト側とこちら側の両方のモデムがMNP等のプロトコルを使用できる場合、モデム―モデム間のフロー制御はモデム同士がプロトコルに従って行うので、特に設定する必要はありません。パソコン―モデム間のフロー制御にはRS/CS制御を使ってください。XON/XOFF制御ではバイナリファイルの転送がうまくいかないことがあるからで

す。TMNの設定もモデムと同様、RS/CS制御にしてください。

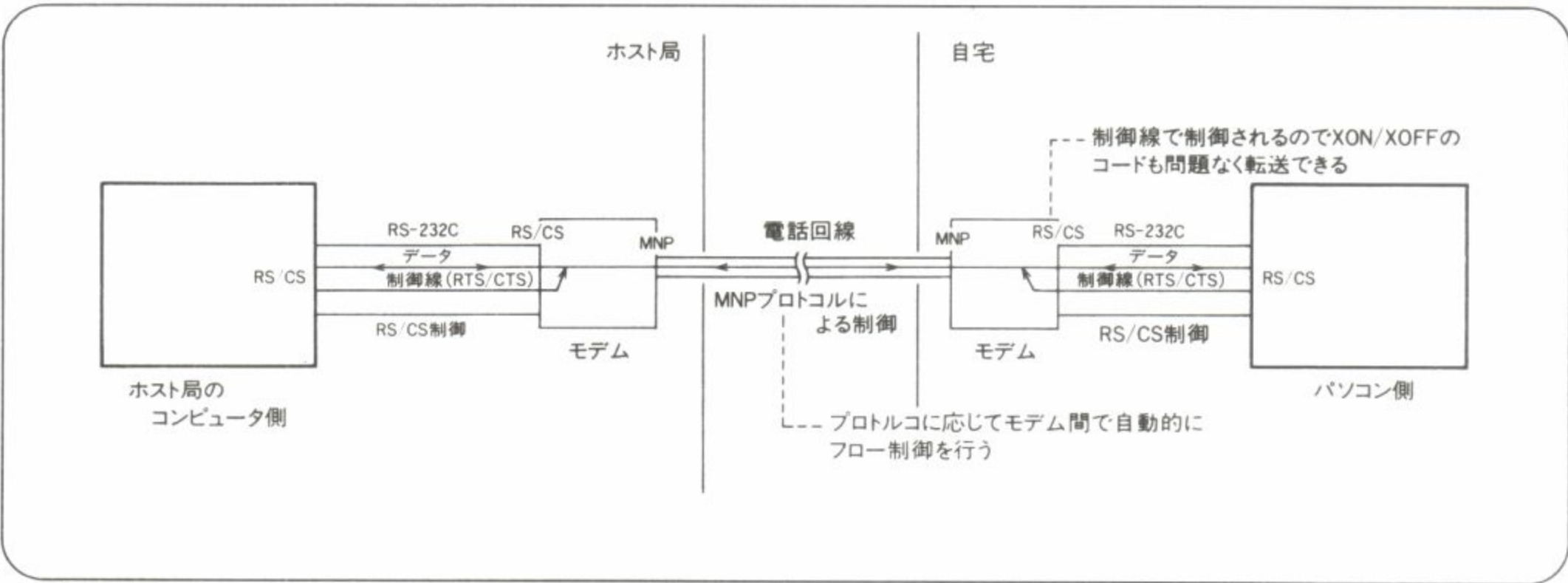
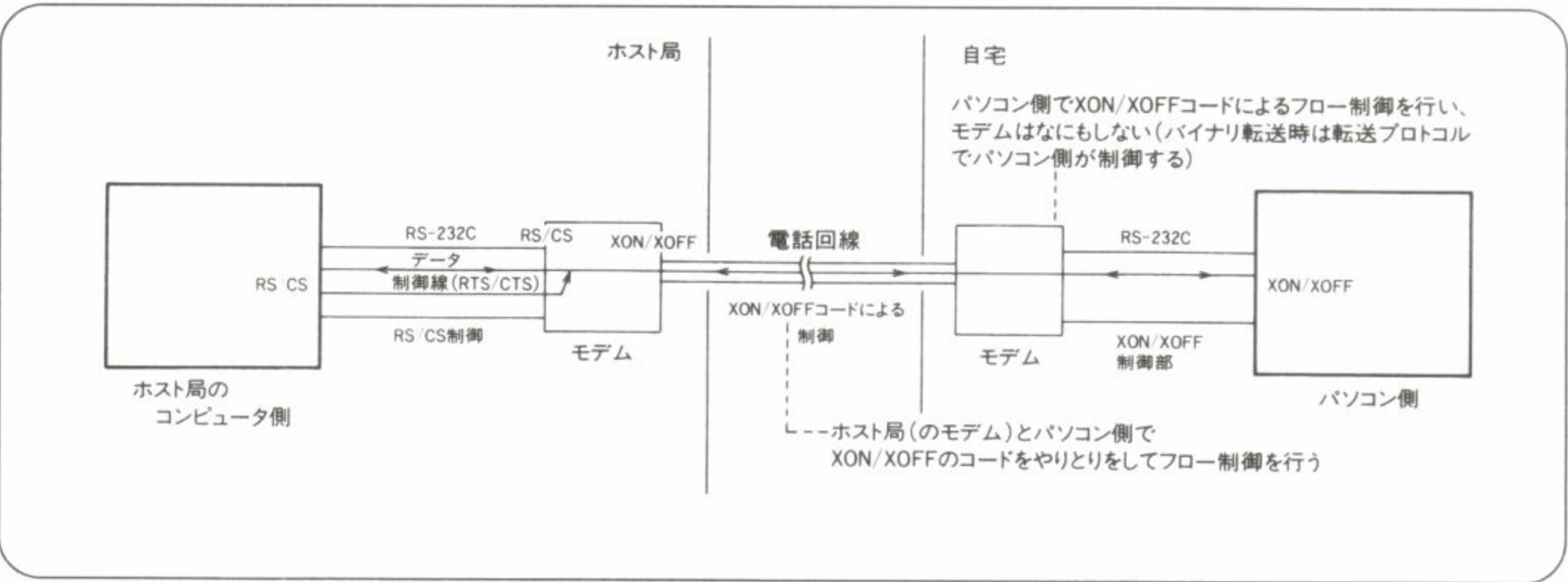
Fig.4 ホストーモデムーモデムーパソコン間のフロー制御(MNP等のプロトコル使用時)

ホスト側のモデムがMNP等のプロトコルを備えていない場合、ホストーパソコン間でXON/XOFF制御が必要になってきます。しかし、モデムーモデム間で独自にXON/XOFF制御を行うと、やはりバイナリ転送のときに問題が生じますので、モデムにはフロー制御はさせずに、TMNの設定をXON/XOFF制御にするだけにとどめておき、端末速度固定モードを解除してRS-232Cの速度を回線上の速度にあわせるようにします。

Fig.5 ホストーモデムーモデムーパソコン間のフロー制御(プロトコル未使用時)

もっとも、最近のホスト局はほとんどMNP等のプロトコルを備えたモデムを使用しているので、RS/CS制御に設定してあれば問題ないはずです。

今まで説明してきたモデムの設定は、対応するコマンドをtmn.cnfのmodem-initに設定すれば、オートダイヤルのときにモデムに送信されます。また、ほとんどのモデムが一度設定したコマンドの内容を電源を切っても保っておけますから、TMNを立ち上げてキーボードからコマンドを打ち込み、モデムに記憶させておけばmodem-initには初期化のコマンドを設定するだけですむこともあります。モデムのマニュアルを参照してください。

具体的にオムロン製のモデムコマンドを説明すると、

ATQ0V1X4¥G0¥J0¥N3¥Q2

V1　：リザルトコードを単語形式にする
X4　：リザルトコードを拡張形式にする(内線電話の場合はX3)
¥G0　：非MNPモード時のモデムーモデム間のフロー制御を行わない

¥J0　：端末速度固定にする
¥N3　：オートリライアブル（プロトコルの自動判定）
¥Q2　：モデム－パソコン間のフロー制御をRS/CS制御にする

その他に、モデムによっては以下のコマンドも設定する必要があるかもしれません。

&C1　：RS-232CのCD信号を、キャリア信号検出時のみON
（接続時間のチェックのため）
&D2　：RS-232CのER信号がONからOFFに変化した場合、回線をOFF
（[SHIFT]＋[CTRL]＋[N]で回線を切るため）
F1　：回線接続時のモデムによるエコーバック禁止
N1　：相手側のモデムに回線速度をあわせる（自動フォールバック）

繰り返しますが、メーカによってコマンドが微妙に違ったり、コマンドではなくディップスイッチで設定する場合もあります。実際にはモデムの取り扱い説明書を参考にしてください。

はじめて通信をする人はモデムの設定方法や用語がよくわからないかと思います。設定を一度に全部理解しようとすると大変です。国産のモデムを使用する場合は、なにも設定しなくても、とりあえずある程度、通信はできるかと思いますので（文字化けしたり、バイナリ転送に失敗したりするかもしれませんが）、とにかく始めてみて問題が起こったら（その場合、モデムか通信ソフト側の設定に問題があることがほとんどです）じっくり勉強してみるのも手かもしれません。

**・ホスト局の選択**

ハードとソフトの設定がすんだら、今度はパソコン通信でアクセスするホスト局を選びます。パソコン通信のホスト局には、大きく分けて(a)大手商用ネットと、(b)草の根ネットの２種類があります。両者の中にもいろいろな特徴・雰囲気を持った、多種多様なホスト局が存在します。いくつかのホスト局にアクセスしてみて、自分にあいそうな雰囲気のネットを選びましょう。

(a) 大手商用ネット

まず、PC-VAN（日本電気㈱が運営するネットワーク）、NIFTY-Serve（ニフティ㈱が運営するネットワーク）などに代表される、企業が事業として経営しているパソコン通信ネットがあります。会員数は多く（30万人を超えるネットもある）、全国からアクセスすることができます。パソコン通信はこれまでも紹介したように、電話回線を使ってホスト局とのやりとりを行います。たとえば、ホスト局が東京にある場合、地方の会員がアクセスすると、電話料金だけでも大変なものになってしまいます。これを避けるため、大手商

用BBSでは、VAN回線を利用したアクセスポイント（APと略したりする。PC-VANやNIFTY-Serveの場合、主な市レベルでの電話回線が用意されている）が全国各地に設けられているため、ホスト局がどこにあっても自宅近くのAPにアクセスすることにより安い電話料金で接続できるので、全国各地の人が多数アクセスしています。そのため、たとえば、北海道の人が沖縄の人と気軽に連絡をとれるようなメリットもあります。ネット内の話題も多岐にわたり、自分と同じ趣味を持った人と話が盛り上がることもあるでしょう。また、商用ネットならではのオンラインショッピング（パソコン通信でデパートなどから通信販売による買い物ができるコーナー）やデータベース検索などのサービスを利用することもできます。

フリーソフトウェアについて見ても、さまざまな機種の、さまざまなソフトウェアが毎日のように新規登録やバージョンアップを繰り返しています。全国各地の有志による作品がネットワークの盛り上がりに一役買っているといってもいいでしょう。

ただし、商用ネットですから、アクセスするにはネットによって定められた方法で行います。たとえば、NIFTY-Serveのイントロパック（市販の通信ソフトやモデムに無料でついていることもある。ユーザID取得に必要な登録をするために、そのネットワークを一時的に利用できるユーザIDやパスワードがついているもの）やオンラインサインアップ（申し込み用紙などを使うのではなく、オンライン、つまり、通信回線上でユーザIDを取得すること）などでIDを取得し、定額あるいはアクセス量に応じた利用料金（課金と呼ぶ）を、電話料金とは別に支払わなければなりません。

(b) 草の根ネット

(a)の大手商用ネットに対して、個人が自分のパソコンなどを使って開いているホスト局があります。これらは一般に「草の根ネット」と呼ばれ、大都市圏を中心として全国各地にさまざまな特色を持ったホスト局が存在しています。会員数も数十〜数百人といった小規模なネットが大部分ですが、その分、大手ネットとは違ったアットホームな雰囲気とまとまりのよさを持ったネットが多いといえます。会員同士のつきあいも深く、ネットワークを離れた実生活でも、共通の趣味を持った集まりとして「オフラインミーティング」と呼ばれる会合やイベント、さらには個人的な友達つきあいも活発です。ネットのムードもSys.op.（BBSの主宰者）と会員の気質を反映してさまざまで、その雰囲気になじむことができればひじょうに心地よいネット活動が続けられるでしょう。

フリーソフトウェアに力を入れているネットや、あるいはパワーユーザと呼ばれるコンピュータのハード・ソフトに精通した会員が多いようなネットワークでは、商用ネットに負けないフリーソフトウェアの質と量を誇っているところも多数存在します。また、大手ネットにフリーソフトウェアをアップロードする前に、作者が内輪の草の根ネットで試験的な公開とバージョンアップを繰り返すような場合も多く見られます。そのテストに参加

することによって自分の意見などを作品に反映してもらうようなチャンスもあるかもしれません。もちろん、草の根ネットだけにしかアップロードされていないような（全国的には）隠れた名作のようなフリーソフトウェアも存在するでしょう。

草の根ネットの多くは、大手商用ネットとは違い、課金をとらず(無料)、IDの取得もオンラインサインアップで気軽に行えるところが大多数です。中にはシステム的な制限から(たとえば、ホストが用意している電話回線数に対して、会員数があまり多いとつながりにくくなるなどの理由で）会員募集を行っていないネットもありますが、その場合でも大抵ゲスト会員(ユーザIDを入力するところで“guest”などと入力すれば、ログインすることができる。ただし、その場合も、他の会員にメイルを出したり、フリーソフトなどがアップされているボードを見ることができないなどの制限がある）としてアクセスすることができます。あちこちのネットをゲストで見て回って、気に入ったところに腰を落ち着けるのもいいかもしれません。

アクセスするには各ネットの情報（アクセス用電話番号など）を調べる必要があります。草の根ネットは、大手ネットと違って広く宣伝活動をしているわけではありませんから、調べるのは大変かもしれませんが、本書の巻末の「BBS一覧」や、全国各地の草の根ネットを紹介した本（電波新聞社から発売されている『ワープロ　パソコン通信BBS電話帳』(2,000円)には、全国の主なBBSの電話番号、特徴などが記載されており、大変便利です）も出版されていますから、最初はそれらを参考にするのもいいでしょう。

また、まわりにパソコン通信をやっている友人がいれば、よく行っているネットを紹介してもらうのも手です。一度パソコン通信を始めてしまえば、ネット仲間からさまざまなネットの情報が自然に入ってくるでしょう。

一般にパソコン通信といえば、大手の商用ネットのみがクローズアップされがちですが、パソコン通信の本当のおもしろさは草の根ネットにあるといっても過言ではありません。草の根ネットの活動に熱中し、自分で新しくホスト局を始めてしまうような人もいます。地方に在住の方は、首都圏のネットにアクセスしようとすると電話代が大変ですが、近場で気のあいそうなネットを探してみるのもいいでしょう。

## 1-6-3 パソコン通信実践編

### 1.NIFTY-Serveの場合

ここまで準備がすんだら、実際に通信を行ってみましょう。

まず、大手商用ネットの例として、NIFTY-ServeのシャープUser's　ForumであるFSHARPにアクセスする場合の手順を示します。NIFTY-Serveには、X68000やMacintosh、PC-9801などのユーザもたくさん集まっています。そこでは、同じような興味を持った人たちが「フォーラム」という集まりを作り、NIFTY-Serveという巨大なネ

ットワークの中に小さなネットワークを作っています（コンピュータ以外の話題も豊富です）。

NIFTY-Serveにアクセスするには、あらかじめNIFTY-ServeのIDとパスワードを取得しておかなければなりません。また、自宅から最も近いAPの電話番号を調べておいてください（NIFTY-Serve関連の書籍を見るなり、ニフティ㈱に問い合わせるなりしてください）。NIFTY-ServeのAPには、ROAD 1からROAD 3の3種類があり、それぞれログインのための手続きが違います。ここではROAD 2の場合の例を説明します。その他の場合はNIFTY-Serveから出されているマニュアル等を参照してください。

通信ソフトTMNを立ち上げたら、まずモデムがきちんと接続され、コマンドを受け付ける状態であることを確認します。キーボードからAT[CR]とタイプしてみてください。

```
AT[CR]
OK
```

このようにOKのプロンプトが返ってくれば、接続はきちんと行われています（“AT”というのが、ATコマンドを打ち込んでいる部分です。ここでは、モデムとの接続がうまくいっているかどうかを確認しています）。返ってこない場合は、ケーブルの接続や通信ソフト（TMN）の設定を確認してください。

次にモデムの初期化（ATZを発行する）を行います。モデムは、単にパソコンのデータと電話回線用のデータを変換するだけでなく、ATコマンドやCCITT V.25bisなどのコマンドを使ってさまざまな設定をモデムに内蔵されている不揮発性メモリに書き込んでおくことができます。ここでは、モデムの初期値はすでにモデム内の不揮発性メモリに記憶されているものとします（はじめてパソコン通信をする場合は、特に設定しておかなくてもかまいません）。

なお、モデムの設定方法や記憶のさせ方については、モデムのマニュアルを参照してください。

```
ATZ[CR]
OK
```

これでモデムは必要な設定値に初期化されました。では、NIFTY-ServeにATコマンドを使って電話をかけてみましょう。ここでは回線はパルス回線（ダイヤル回線）とします。トーン回線（プッシュ回線）の場合は、コマンドのATDPの部分をATDTと入力してください。ここで用いた電話番号755-4466は、川崎AP、ROAD 2の例です（1993年1月現在）。

ATDP755-4466[CR]

モデムから電話をかけている「カタカタ、カタカタ」という小さなパルス音が聞こえてくるはずです。モデムのボリュームがオフになっていなければ（そして、回線がふさがっていなければ）、続いて呼び出し音と電話をとる音、そして「ピー、ガガッ」という独特のノイズのような音が聞こえてきます。これはホスト側のモデムが電話に出たことを示しています。しばらくすると（モデムの設定がうまくいっていれば）、こちらのモデムが応答し、画面に以下のような回線の接続を示すメッセージが出るはずです。このメッセージも、モデムの設定によっていろいろ変わってきます。

CONNECT 2400/REL

ここでAPに接続するための“@P”（またはスペースと大文字の“P”）を入力します。これは画面に表示（エコーバック）されません。なにも表示されなくても気にせずタイプしてください。

@P[CR]　　（画面には表示されない）

続いてNIFTY-Serveが使用しているVAN回線FENICSに接続したことを示すメッセージとホスト名の入力を求めるプロンプトが出ます。

FENICS-ROAD 2
HOST NAME?
＊

ここで＊に続いてNIFTY-Serveのホスト局につなぐことを示すコマンドを入力します。

＊C NIF[CR]

これでNIFTY-Serveに接続されます。続いてNIFTY-Serve内の接続先を示すコネクションIDを聞いてきます。通常は“SVC”と入力します。

COM

Enter Connection-ID　--->SVC[CR]

COLUMN

**イントロパックでのID取得**

イントロパックでログインする場合は、イントロパックに書いてある指示に従ってコネクションIDをSVCのかわりに“SGN”と入力した後、Serial-No.とAgreement-No.を入力します。

Enter Serial #--->
Enter Agreement #--->

これは、NIFTY-ServeのID取得のためのもので、手続きがすむと一時利用のためのIDとパスワードが交付されます（正式のIDは後日郵送されます）。

次に、あなたのユーザIDとパスワードを入力します。つまり、ログインする相手があなたであることをホスト側に知らせるわけです。なお、パスワードは機密保持のため、画面に表示されません。ここではIDがABC01234、パスワードがZYXWVUの例です。

パスワードは、最初はホスト側で用意したものが定められていますが、最初にログインしたら変更してください(変更する場合は、NIFTY-Serveの最初に表示されるTOPメニューの「1.サービス案内」の中から「12.会員情報」を選んでください)。パスワードは、なるべく複雑な文字の並びにしましょう。自分の名前や今回の例のように容易に推測できるようなものはだめです。ログインに成功すると、NIFTY-Serveからのメッセージが出力されます。

```
 Enter User-ID  --->ABC01234[CR]        (あなたのIDを入力)
 Enter Password --->ZYXWVU[CR]          (パスワードを入力。画面に表示されない)

ようこそNIFTY-Serveへ
  Copyright (C) 1993
 by NIFTY Corporation
 All Rights Reserved

〔定期メンテナンスのお知らせ〕
  定期メンテナンスのため、 1月20日（水）午前 2時00分から午前 9時00分まで、
センターのサービスを停止させていただきます。
  会員の皆様には、大変ご迷惑をお掛け致しますが、よろしくお願い申し上げます。

前回LOG OUT 93/01/17    23:24:15


NIFTY-Serve     TOP
 1. サービス案内・検索              2. 電子メール
 3. 掲示板                          4. CBシミュレーター
 5. フォーラム                      6. ニュース／スポーツ／天気予報
 7. 企業／経済／ビジネス            8. 新聞・雑誌記事情報
 9. 生活／教育／就職               10. 趣味／旅行／ゲーム
11. ワープロ／コンピューター       12. ショッピング／アドコーナー
13. 海外データベース(INFOCUE)      14. コンピュサーブコーナー
15. ビギナーズコーナー              E. 終了
>
```

この＞（プロンプト）が出た状態がNIFTY-Serveの（メニューモードでの）コマンド入力待ちの状態です。この状態でNIFTY-Serveのコマンドを入力します。

では、シャープUser'sフォーラムであるFSHARPに行ってみましょう。FSHARPは「ユーザー館」FSHARP1と、「アミューズメント館」FSHARP2、および「ワークステーション館」FSHARP3の３つに分かれています（1993年１月現在）。FSHARP1はX68000を中心としたシャープ一般の話題、FSHARP2はゲーム、CG、音楽に関する話題、FSHARP3はプログラミングを中心とした、パワーユーザ向けの話題を扱うフォーラムです。

ここでは、まずFSHARP1に行ってみましょう。フォーラムに直接入るにはGOコマンドを使用します（TOPメニューから順に選んでいくこともできますが、GOコマンドを使って直接行ったほうがてっとり早いといえます）。

＞GO FSHARP1[CR]

最初にフォーラムに入る場合は、そのフォーラムの入会手続きを行います。入会手続きでは、そのフォーラム内で使用する名前（ハンドル名と呼ばれる）を聞いてきます。もちろん、本名でもいいのですが、せっかくなのでなにか気のきいた名前を考えておくといいかもしれません。ここでは、ありきたりですが、「ぺけろく」という名前にしてみましょう。

名前を登録し終わると入会手続きが完了し、フォーラムに入ります。

```
<シャープUser'sフォーラム>     FSHARP
1:フォーラム概要    2:一時利用    3:入会手続き
>3[CR]
氏名（漢字で8文字以内 改行のみ実名）
:ぺけろく[CR]
確認　ぺけろく          (1:OK  2:NG)
:1[CR]
登録（1:登録する　2:しない）
:1[CR]
ー登録完了ー

___________ FSHARP1 Information 93-01-18 08:30:00 ___________

 1/18  TMNUPD            (TMN Final versionのバグを修正する) 登録  (LIB 1)
 1/18  TMNFEE v1.10 + η (TMN用電話料金集計プログラム      ) 改版  (LIB 1)

                      ・・・（途中省略）・・・

12/07  SFC       v1.1   (ｽｰﾊﾟｰﾌｧﾐｺﾝのｺﾝﾄﾛｰﾗｰをX68kに接続する ) 改版  (LIB 2)
12/07  OBM       v1.00  (メニューソフトウェア『OBMenu』    ) 登録  (LIB 2)

 11月30日までのインフォメーションは「お知らせ（ANN)」に登録されています。
______________________________________________________________________

  御入会ありがとうございます。

  あなたは現在準会員です。入会承認はまだ行っておりませんが、正会員と
  同様のサービスを受けることができます。

  電子会議に行かれましたら、まず11番のボードに自己紹介を書いて下さい。

  ※）当フォーラムでは、掲示板機能を公開していません。

<SHARP User's Forum ユーザー館>    FSHARP1
1:お知らせ            *:掲示板      3:電子会議
4:データライブラリ    5:会員情報    6:リアルタイム会議
7:SYSOP 宛メール      8:オプション  E:終了
>
```

フォーラムに入るとフォーラム内のメニューが出ます。とりあえずメッセージにあったように自己紹介を書き込んでみましょう。メニュー3番の電子会議を選ぶと、会議室の一覧が出ます。

```
>3[CR]
番号  発言 (未読)   最新    会議室名
 1     348 (  10)   01/24   ふりーとーく        -9-
 2     999 ( 999)   08/27   PDSサロン           -6-

                    ・・・（途中省略）・・・

10     122 (  10)   01/20   MZシリーズの部屋    -3-
11     141 ( 141)   01/22   自己紹介してね      -4-
12     129 (  10)   01/22   SHARP製品の部屋     -1-

                    ・・・（途中省略）・・・

19     747 ( 747)   01/24   PDSサロン           -7-
20     999 ( 999)   07/04   自己紹介してね      -3-   /* Closed */
```

ここでは11番の自己紹介用の会議室を選びます。

```
>11 [CR]
電子会議 (1:発言　改行のみ: 読む) 通常モード
>
```

これで電子会議のコーナー（通常「ボード」と呼ばれます）に入りました。

では、書き込んでみましょう。１の「発言」を選びます。誰かの発言に対するコメントではないので、属性は「メッセージ」です。ここではオンラインで直接キーボードから書き込みをしていますが、あらかじめエディタなどで書いておいた文章を、TMNの機能を使ってオートタイプ（自動送信）するのもいいでしょう（詳しくはTMNのページを参照してください）。

```
電子会議 (1:発言　改行のみ: 読む) 通常モード
>1 [CR]
属性 (1:メッセージ　2:コメント)
 : 1 [CR]
本文 (300 行まで　終了は行頭で/E)
[CR]
    はじめまして、ぺけろくといいます。[CR]
[CR]
    パソコン通信は初心者です。どうぞよろしくお願いします。[CR]
[CR]
[CR]
                              ぺけろく [CR]
/E [CR]
修正 (1:修正する　2:しない)
 :
```

本文を書き込み中の修正は[BS]キーによる後退のみでしか行えません。カーソルキーなどは使えません（一見、うまく修正されたように見えても、送信された内容は判読のできない記号や文字だらけになっています）。そして、一度改行の[CR]キーを押してしまうと、もう前の行の修正はできません。いったん、その書き込みをやめて最初から書き直すか、または、書き込み終了後、今書いた文章を修正するかどうかを聞かれるので、そのとき修正するしかありません。そういう意味でも、初心者はあらかじめ書き込む内容を用意しておいたほうがいいでしょう。

ここでは、書き込みの例を示すため、一言で自己紹介をすませてしまいましたが、あらかじめある程度詳しいプロフィールを用意しておいたほうがいいでしょう。

なお、強制というわけではありませんが、一般に書き込みの最後には自分のハンドル名（署名）を書き込みます。うまく書き込めたらタイトルを入れて登録します。

```
修正（1:修正する　2:しない）
：2[CR]
題名（漢字で20文字まで）
：初めまして、べけろくといいます。[CR]
確認　初めまして、べけろくといいます。（1:OK　2:NG）
：1[CR]
登録（1:登録する　2:しない）
：1[CR]
ー登録完了ー
>
```

これで書き込みが完了しました。

では次に、フリーソフトウェアが登録されているデータライブラリに行ってみましょう。データライブラリはフォーラム用メニューの4番で入ることができます。メニュー以外の場所から直接移動するにはLIBコマンドを使用します。

```
>LIB[CR]

  ☆ゲーム，ＣＧ，ＣＧＡ，音楽関連のソフトウエア、データは
    FSHARP2 『シャープUser'sフォーラム・アミューズメント館』へ移動しました。

  ☆SX-WINDOW, KO-WINDOW, プログラム解析，開発，ＴｅＸ関連のソフトウエアは
    FSHARP3 『シャープUser'sフォーラム・ワークステーション館』へ移動しました。


番号  総数  登録済  最新   ライブラリ名
   1   100  (  97)  01/18  Ｘ６８０００【通信ソフトとアーカイバ】
   2   332  ( 307)  01/18  Ｘ６８０００【各種プログラム】
   5    43  (  34)  10/10  各種データ
   6    23  (  14)  03/05  Ｘ１シリーズ
   7    34  (  19)  07/26  ＭＺシリーズ
   8    24  (  22)  01/06  CP/M80 & MS-DOS（機種依存性の無い物）
   9    82  (  82)  03/03  過去の会議室とその議事録
 >
```

データライブラリは、プログラムやデータの内容ごとに分類され登録されています。ここでは、2のX68000用各種プログラムのライブラリに入ってデータ一覧を覗いてみましょう。

```
>2[CR]
データライブラリ（1:データ一覧　2:検索　3:ｱｯﾌﾟﾛｰﾄﾞ（無料）　4:ﾀﾞｳﾝﾛｰﾄﾞ　E:終了）
>1[CR]
番号 ID       登録日付 バイト  参照   データ名
907 NAH00720 93/01/09  61133   24 T とぅぇにぃわん Ver 1.23
906 NAH00720 93/01/09  63569   24 T fsck.x ver 0.05
904 NCC02715 93/01/07   1261   10 T ディスクバッファ消去 BUFFCLR.X
902 MHH01343 93/01/06 110217   12 T MF.X G2.05α
900 MHF03052 93/01/04  55711   29 T バイナリファイルエディタ【FES】v1.01g
896 MHH01343 92/12/29  71946  447 T BdifSet.ish
894 GCF00341 92/12/28  12057   95 T テキストファイル分割ツール「TCP.x」

                    ・・・（途中省略）・・・

879 HFD00043 92/11/15   5248  574 T X68K System-Information Ver 1.02
878 MHH01343 92/11/15  53201  687 T Bdif v1.1rel7 & Bup v1.1rel12
番号（改行で次頁）
>
```

画面に1ページ分の登録されているプログラム一覧が表示されます（初期設定では24

行。変更するには、TOPメニューから、「1．サービス案内・検索」→「12．会員情報」(表示／変更)→「3：端末設定」→「1：画面行数」で変更することができます)。さらに続きが見たいときは[CR]キーのみを入力します。また、この中に欲しいプログラムがあった場合は、その番号を選んでダウンロードします。たとえば、896番のBdifSetをダウンロードする場合を例に挙げてみます（実際には、このプログラムは添付ディスクの中に入っているのでダウンロードする必要はありませんが、NIFTY-Serveでのフリーソフトウェアのダウンロードのしかたの例として説明しておきます)。

```
番号（改行で次頁）
>896[CR]
データ名：BdifSet.ish
      ID：MHH01343
登録日付：92/12/29
    属性：テキスト
  バイト：71946
    参照：445
補足説明：

--------------------------------------------------------------------
【プログラム名】BdifSet.ish
【制    作  者】ひがしで
【転載 の 可否】可（ドキュメント参照）
【動 作  環 境】Human68k
【展 開  方 法】ISH(MIC) -> LHA
【使 用  方 法】ドキュメント参照
--------------------------------------------------------------------

                           BdifSet.lzh

                        ・・・（途中省略）・・・


処理（1:ダウンロード  2:表示  E:終了）
>
```

ダウンロードしたい番号を選ぶと、最初にそのプログラムの名前、登録者のID番号、登録された日付、登録されているファイルの属性、ファイルのバイト数、何人の人によってダウンロード（参照）されたかなどが表示されます。

ちなみに、ここでいうファイルの属性とは、そのファイルがどのような形式で登録されているかを表し、テキストとバイナリの２種類があります。FSHARPの場合は必ずテキスト形式で登録されています。他のフォーラムの場合、プログラムの登録はバイナリ属性の場合が多いかもしれません（バイナリ属性で登録されている場合は、処理の「2：表示」は使えません。当然、メニューにも出てきません)。この２つの違いは、おおむね以下のような点です。

(a) バイナリ形式

- プログラムが、通常LHAなどで圧縮された形式でそのまま登録されている。したがって、表示できない（COMMAND.X上で拡張子が“.X”の実行ファイルがTYPEできないようなもの)。

・そのため、ホストから転送してくるためには特別な各種のプロトコルを使う必要があり、通信ソフトまたは外部プログラムによるサポートが必要になる。
・用いるプロトコルによっては、元のファイルのファイル名やタイムスタンプが正確に再現できないことがある。
・プログラムにデータ変換等がかかっていないので、冗長部分がなく、転送するファイルサイズが大きくならない。

(b) テキスト形式

・プログラムに変換(バイナリ→テキスト変換)をかけ、画面表示可能なファイルにしたものが登録されている。
・したがって、ホストから転送してくるために特別なプロトコルは必要なく、通信ソフトのログ保存を行うこと(画面に表示されたものをすべてファイルに保存しておくこと)でプログラムを手に入れることができる。前に紹介したishを使えば、ログファイルの中からishを使ってテキスト変換した部分だけを元のファイル(通常LHAなどで圧縮されている)に戻すことができる。
・変換前の元プログラムのファイル名やタイムスタンプが保存される。
・バイナリ形式の元プログラムをテキスト形式に変換することや、通信時の文字化けによるエラー訂正を行えるようにするために、転送ファイルサイズが元プログラムよりも大きくなる。

ファイル属性の違いがおわかりいただけたでしょうか。

続けて補足説明が表示され、プログラムの内容を説明します。1画面表示するごとに確認の[CR]入力を求めてきます(この機能は途中でやめることもできます)。最後に「ダウンロード」または「表示」するか、あるいは「終了」するかを聞いてきます。

ここではプログラムを転送するために1のダウンロードを選択します。

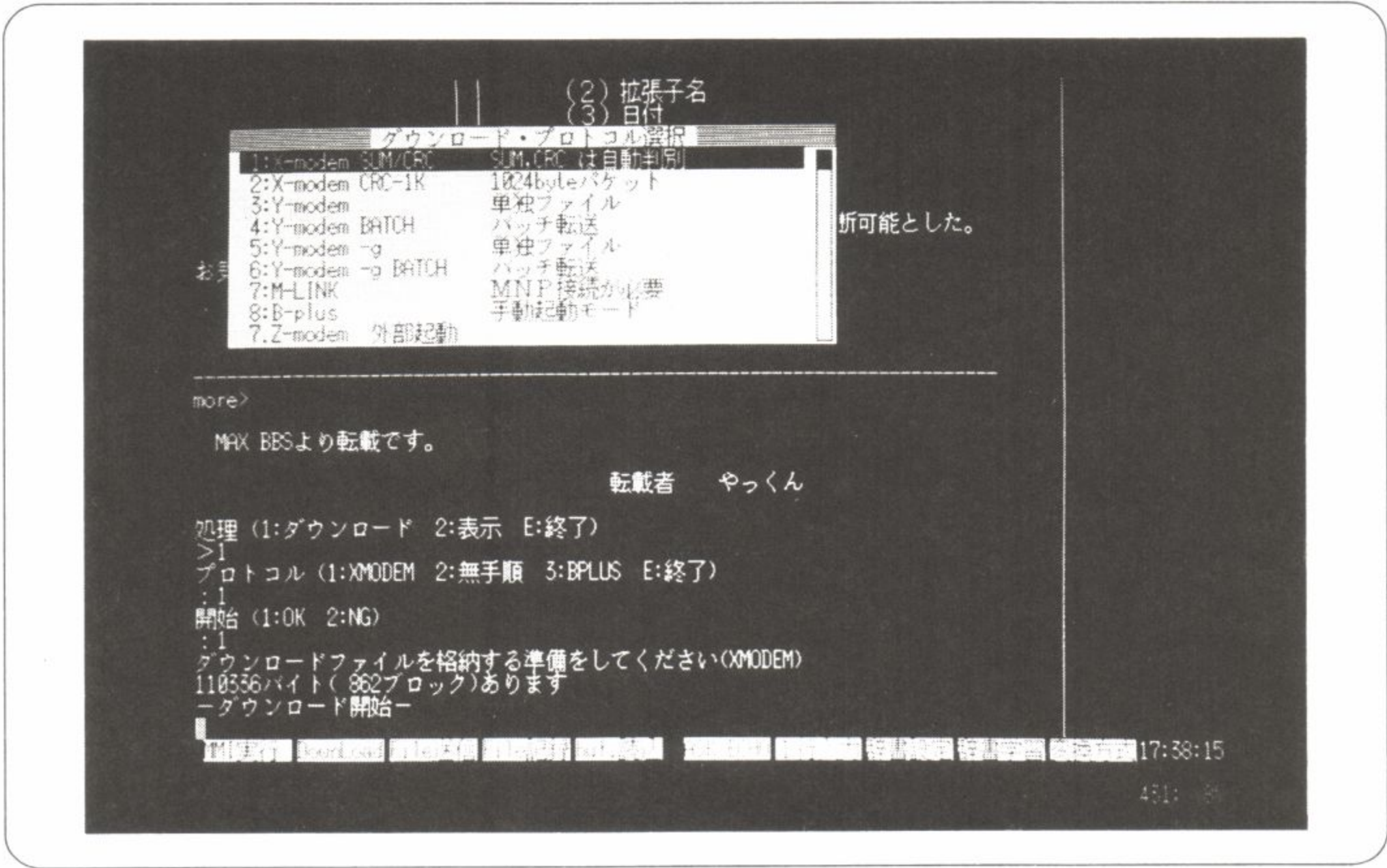
```
処理 (1:ダウンロード  2:表示  E:終了)
>1[CR]
プロトコル (1:XMODEM  2:無手順  3:BPLUS  E:終了)
:
```

すると今度は、転送に用いるためのプロトコルを聞いてきます。プロトコルはデータ(テキストファイルやバイナリファイルなどのファイルを含む)を転送するためのホスト−ターミナル間の手続きのことです。NIFTY-Serveでは無手順(「処理」の「2:表示」と同じです。ただし、改行コードの変換は行われません)の他に、XMODEMとBPLUSという2つのプロトコルがサポートされています。プロトコルには、この他にもYMODEM、ZMODEM、Quick-VAN、M-LINKなどさまざまなものがあり、ネットによってサポートされているプロトコルが違います。

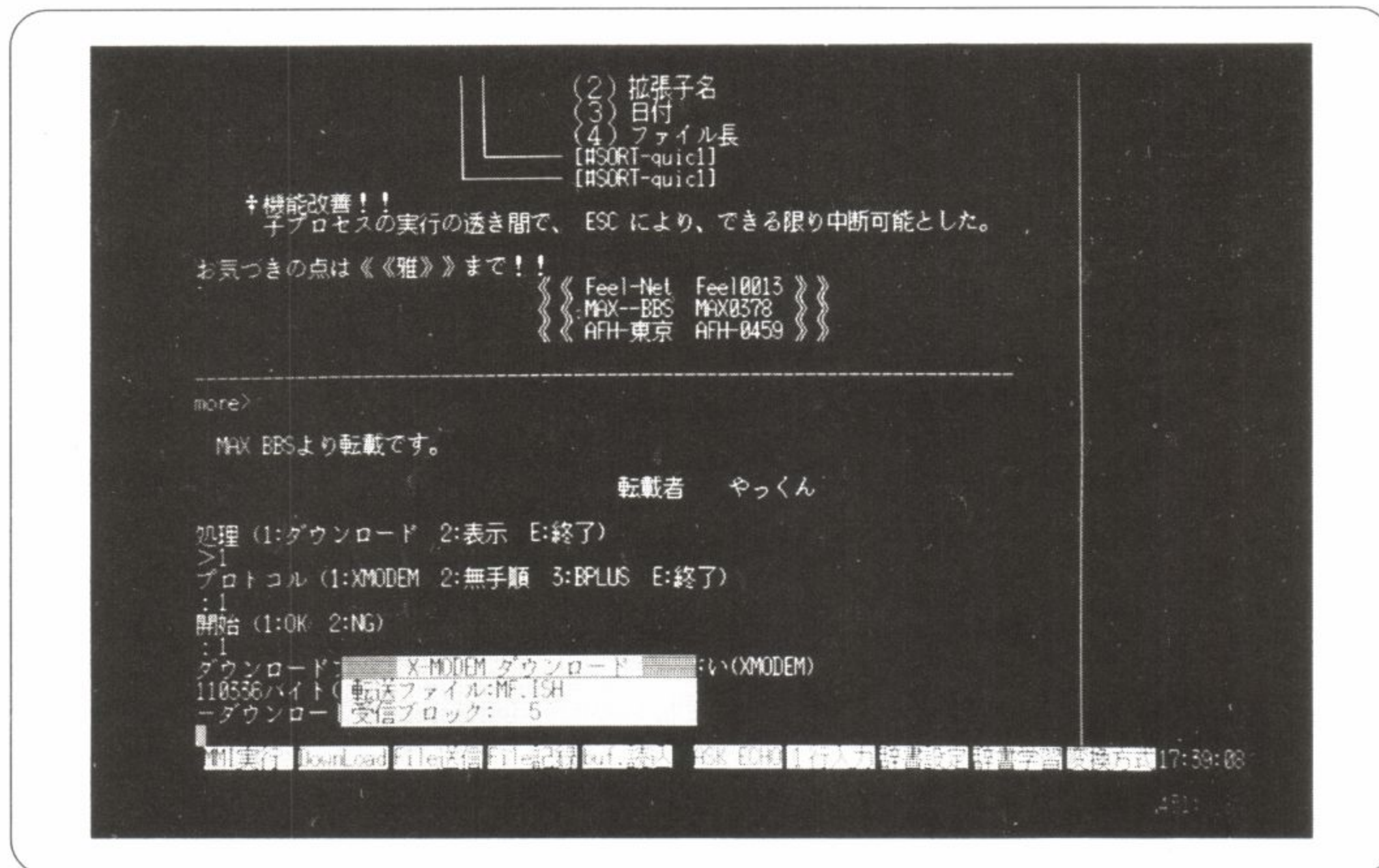
ここでは最も多くのホスト局でサポートされていると思われる、メニュー番号１のXMODEMを使用してみましょう。ただし、XMODEMはMNPのモデムとひじょうに相性が悪く、転送速度が極端に落ちるため、通信ソフトがBPLUSを使用できる場合は、NIFTY-ServeではBPLUSを使ったほうがいいでしょう。TMNは、どちらのプロトコルもサポートしています。

```
プロトコル (1:XMODEM  2:無手順  3:BPLUS  E:終了)
 : 1 [CR]
開始 (1:OK  2:NG)
 : 1 [CR]
ダウンロードファイルを格納する準備をしてください(XMODEM)
 72064バイト( 563ブロック)あります
ーダウンロード開始ー
```

開始と聞かれて１の「OK」を選ぶと、転送するプログラムの大きさをバイト数とXMODEMで使用するブロック数で表示し、ダウンロードを開始します。「ーダウンロード開始ー」と表示されたら、ホスト側は転送状態になっているので、通信ソフト側でも転送の処理を開始します。DownLoadのキー（TMNの初期設定では[F2]キー）を押してプロトコルの一覧を表示し、１番目のX-modem　SUM/CRCを選んで[CR]キーを押します(Pht.5参照)。

Pht.5　TMNのプロトコル一覧

すると、セーブするためのファイル名を聞いてきます。ここでは“BdifSet.ish”と入力します。ファイル名を入力して[CR]キーを押すと、転送状態を表示するウィンドウが開いて途中経過を表示します（Pht.6参照)。

Pht.6　転送状態表示ウィンドウ

```
                  (2) 拡張子名
                  (3) 日付
                  (4) ファイル長
                  [#SORT-quicl]
                  [#SORT-quicl]
†機能改善！！
  子プロセスの実行の透き間で、 ESC により、できる限り中断可能とした。
お気づきの点は《《雅》》まで！！
                  《《 Feel-Net  Feel0013 》》
                  《《 MAX--BBS  MAX0378  》》
                  《《 AFH-東京  AFH-0459 》》
-----------------------------------------------------------------
more>
  MAX BBSより転載です。
                              転載者　　やっくん
処理 (1:ダウンロード 2:表示 E:終了)
>1
プロトコル (1:XMODEM 2:無手順 3:BPLUS E:終了)
:1
開始 (1:OK 2:NG)
:1
ダウンロード   X-MODEM ダウンロード    い(XMODEM)
110556バイト  転送ファイル:MF.ISH
-ダウンロー   受信ブロック:    5
```

転送が終了するとウィンドウを閉じ、メッセージウィンドウで転送終了のメッセージを表示します。これで転送は終了です。

画面には、以下のように、ホスト側の転送終了を知らせるメッセージが表示されているはずです。

```
ーダウンロード終了ー
データライブラリ (1:データ一覧　2:検索　3:ｱｯﾌﾟﾛｰﾄﾞ (無料)　4:ﾀﾞｳﾝﾛｰﾄﾞ　E:終了)
>
```

では、NIFTY-Serveを終了し、今ダウンロードしたプログラムがちゃんとディスクの中にあるかどうか確認してみましょう。NIFTY-Serveとの接続を切る(ログオフする)ためには“OFF”というコマンドを入力します。

```
データライブラリ (1:データ一覧　2:検索　3:ｱｯﾌﾟﾛｰﾄﾞ (無料)　4:ﾀﾞｳﾝﾛｰﾄﾞ　E:終了)
>OFF[CR]

        LOG IN  ---  93/01/18    20:18:23
        LOG OUT ---  93/01/18    20:20:35

  ご利用時間は、　02分12秒でした。
  ご利用誠にありがとうございました。
```

NIFTY-Serveを終了すると、最初のFENICSの接続画面になるので、HOST NAME?の問いに“OFF”と入力して電話回線を切ります(ここで、最初のように“C NIF”と入力すれば、再びNIFTY-Serveに入ることもできます)。

```
Clear PAD
Host requested clearing the call

HOST NAME?
*OFF [CR]

NO CARRIER
```

モデムから“NO CARRIER”のメッセージが出力されたら、ホストとの電話接続が切れたことになりますので、TMNを終了してください。[SHIFT]+[CTRL]+[Q]の3つのキーをいっしょに押します。

これでNIFTY-Serveからプログラムをダウンロードできました。転送時に入力したディレクトリにファイル（ここではBdifSet.ish）があることを確認してみてください。

今回の例では、フリーソフトをダウンロードするために必要な、最低限のNIFTY-Serveのコマンドしか説明しませんでしたが、実際にはさまざまなコマンドがあります。NIFTY-Serveのマニュアルやオンラインヘルプ等を参考にしてください。

なお、FSHARPではプログラムには必ずishによるテキスト変換をかけていますので、実際に使用する前には添付ディスクに入っているishで逆変換してやる必要があります。具体的にはish が実行できる状態で

C>ish BdifSet.ish[CR]

と入力します。変換が行われ、BdifSet.Lzhという圧縮ファイルができるはずです。先ほど説明した要領でLHAを使って展開します。

**2.MAX BBSの場合**

次に、草の根ネットへのアクセス例として、MAX BBSという、筆者たちがよくアクセスするネットに接続してみましょう。

NIFTY-Serveの場合と同様にTMNを立ち上げ、モデムを初期化したあと、MAX BBSにダイヤルします。MAX BBSに接続されると、画面に“CONNECT”と表示され、続いてMAX BBSのオープニングメッセージが表示されます。

```
AT[CR]
OK
ATZ[CR]
OK
ATDP03-5707-6720[CR]
CONNECT 2400/REL
==============================

              M  A  X  B  B  S

     System       : X68000 PRO HD / 80MB HDD
     Host program : HOST PRO68K 9 ver 1.15.05.32

     channel 0      : Host
     channel 1      : 03-5707-6720 (代表)
     channel 2,3    : 03-5707-6721,2
     channel 3      : 03-5707-6723 (9600bps V.32&V.42bis)
==============================
```

すると、接続したチャネル（回線）番号を表示したあと、ユーザIDの入力を求めてきます。最初は“GUEST”でログインします。GUEST[CR]とすると、ログネーム(ハンドル名）を聞いてくるので、あなたのハンドル名を決め、入力してください。そのあと、いろいろなメッセージを表示してコマンド待ちになります。

```
[System] ## 接続完了 1ch ##

GUEST id = "GUEST"
Your ID  :GUEST[CR]

[ 半角８文字以内でログネームを入力してください ]:ぺけろく[CR]

## 現在のあなたのレベルは【お客様】です。

## 開局通算            : 139987人目
## 本日通算            : 122人目

## 前回ログアウト日時 : 93/01/18 00:00:00
## 今回ログイン日時   : 93/01/18 17:54:24



#####################################################
#                                                   #
# ＭＺとＸ１とＸ６８０００の集まりとして始められた #
# ＢＢＳです。マイコンクラブ「全国おらがＭＺ」の同 #
# 志たちが集まって、できたＢＢＳです。ホストはＸ６ #
# ８Ｋで２５とＸ１がつながれています。ほかに集会も #
# 月一度以上行っていますので是非参加してください   #
#                                                   #
#####################################################



[System] ## アクセス時間は 10 分です ##

 ボードに新しい書き込みがあります
#0 (#SYS  )   #1 (#FREE )     #8 (#SHOP )     #9 (#TSHOP)

Root [?:Help  e:Logoff]:
```

この状態で使用できるコマンドは“?”で表示されます。ここでは会員登録を行うため、Iコマンドを使用します。

```
Root [?:Help  e:Logoff]:?[CR]

B    : 電子掲示板      (各ジャンル別に用意してあります)
M    : 電子手紙        (個人的な話題の場合はこちらを利用します)
C    : 電子会議室      (チャット・アクセス者一覧表など)
I    : 会員情報        (会員登録方法・変更・一覧表など)
U    : 電子掲示板設定  (自動読み出し対象掲示板の設定)
A    : 全ボードの新しい書き込みを読む(ノンストップモード)
S    : 全ボードの新しい書き込みを読む(ストップモード)
E    : 終了            (アクセスを終了します)
?    : ＨＥＬＰ表示    (ＨＥＬＰメッセージを表示します)
H    : ＨＥＬＰ表示    (ＨＥＬＰメッセージの表示を"ON"/"OFF"します)
HELP : 詳しい説明を表示します

Root [?:Help  e:Logoff]:I[CR]

Profile [e:終了  ?:Help]:
```

ここでも使用できるコマンドは同様に“?”で表示されます。“W”コマンドで会員になるための入会手続きを行います。

```
Profile [e:終了  ?:Help]:?[CR]

L : 会員一覧表の表示
R : 会員情報の表示
W : 入会手続き (ゲストのみ)
C : 会員情報変更
S : 会員情報検索
E : 終了

Profile [e:終了  ?:Help]:W[CR]

          <<<<  ＭＡＸＢＢＳ入会のご案内  >>>>

入会には次の９項目が必要です。

      1.パスワード      : 半角の英数字８文字まで
      2.本名
      3.ログネーム
      4.年齢
      5.電話番号         市外局番も含めてすべて (ex:03-474-7237)
      6.職業
      7.住所
      8.自己紹介
      9.システム構成

すべてを入力すると、ID番号が表示されます。そのあと、SYSOP宛に入会希望のメール
を発送してください（その際、このBBSをどこで知ったのかも書いてもらえれば幸いで
す）。
なお、ここで登録されたIDは次回のログインから有効となります。
この状態では会員レベルは【ゲスト】となっています。

ＩＤ登録内容とメールを確認次第、レベル変更いたします。


※ スタッフ一同、貴方の入会を心より御待ち致してしております。
```

“Y”と答えて、入会のための各種項目を入力します。順に答えていってください。

```
入会手続きを行いますか [Y/n]:Y[CR]
パスワード       = ZYXWVUTS[CR]
本名             = 山田　太一郎[CR]
ログネーム       = ぺけろく[CR]
年齢             = 20[CR]
電話             = 03-1234-5678[CR]
職業             = 東盟大学学生[CR]
住所             = 東京都千代田区外神田１丁目２番地３号[CR]
自己紹介         = ぺけろくといいます。どうぞよろしく。[CR]
システム構成     = X68000XVI + 180M HDD[CR]

パスワード       = ZYXWVUTS
本名             = 山田　太一郎
ログネーム       = ぺけろく
年齢             = 20
電話             = 03-1234-5678
職業             = 東盟大学学生
住所             = 東京都千代田区外神田１丁目２番地３号
自己紹介         = ぺけろくといいます。どうぞよろしく。
システム構成     = X68000XVI + 180M HDD

この登録に間違いないですか [Y/n]:
```

入力に間違いがなければ、“Y”を入力して終了します。間違っていたら“N”を入力してやり直します。言うまでもありませんが、本名や電話番号は正しく書くようにしてください。会員登録が終了すると、あなたのID番号が表示されますので覚えておいてください。次回からは、このID番号と先ほど入力したパスワードでログインを行います。

```
この登録に間違いないですか [Y/n]:Y[CR]
[System] ## ＩＤを検索中。しばらくお待ちください ##
[System] ## ＩＤは "MAX0347" です ##
Profile [e:終了  ?:Help]:E[CR]
Root [?:Help  e:Logoff]:
```

登録のための入力が終了したら、先ほどのメッセージにあったようにSys.op.宛にメイルを出します。メイルの発送は、Root（最初に表示されたプロンプト）から“M”コマンドで行います。ここで使えるコマンドは、やはり“?”で表示されます。“W”コマンドでメイルを出します。宛先はSys.op.なので１番です。

```
Root [?:Help  e:Logoff]:M[CR]

Mail Box [r:read  e:end  ?:Help]:?[CR]

R  : 手紙を読む（読み終えた手紙は早目に削除しましょう）
W  : 手紙を書く（最大１６人まで）ゲストは SYSOP宛のみ
L  : 手紙到着一覧表を表示する
P  : 手紙送信一覧表を表示する
D  : 送った手紙を削除する
E  : 終了

Mail Box [r:read  e:end  ?:Help]:W[CR]

宛先ＩＤ [ 終了 "." or "END" ]:1[CR]

MAX0001[Sys.Op] さんですね [Y,n,e]:Y[CR]

宛先ＩＤ [ 終了 "." or "END" ]:.[CR]

MAX0001[Sys.Op] さんに発送します。

        ============================================
         書き込みの終了は改行後の行頭で"END"か"."です
          (519168 バイト までの書き込みが可能です)
        ============================================

 # title = ______________________________
```

タイトルに続けて本文を入力します。方法は、NIFTY-Serveの場合とほぼ同じで、やはり修正する場合は[BS]キーのみです。入力の終了は、行の先頭で“END”または“.”です。なお、１行の入力は、画面の上に表示されている80桁のカラム位置を超えないようにするのがエチケットです。X68000でTMNを使って通信している場合、横は96桁まで表示可能ですが、多くの機種では横幅は80桁までしか表示できないため、それを超えると中途半端なところで改行され、ひじょうに見づらい文章になってしまうからです。これは、MAX BBSだけでなく、どこのBBSでも同様のマナーです。最後に、今回の書き込みバイト数が表示され、書き込みを示す“W”を入力すると、メイルが発送されます。

```
 # title = ぺけろくです。よろしく！[CR]

----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+---- 8

[CR]
ぺけろくといいます。フリーソフトウェアの本を見てやってきました。[CR]
[CR]
どうぞよろしくお願いします。[CR]
[CR]
[CR]
                                          ぺけろく[CR]
[CR]
.[CR]
( Received data: 150 byte)

[w:書き込み  v:確認  a:追加  q:中止]:W[CR]

MAX0001[Sys.Op] さんへ送っています ... 終了

Mail Box [r:read  e:end  ?:Help]:
```

これで登録のための手続きがすべて終了しました。いったんMAX BBSを抜け、登録を待ちます。特に問題がなければ、数日のうちにID登録が行われ、ログインできるようになるはずです。

```
Mail Box [r:read  e:end  ?:Help]:E[CR]

Root [?:Help  e:Logoff]:E[CR]

ログアウトします。よろしいですか？ [y/N]:Y[CR]

[System] ## 是非、会員になって活躍してくださいね ##

   Login at: 93/01/18 17:54:24
   Now     : 93/01/18 18:03:44
   Used    : 00:09:38 ( 578sec )

 ご利用いただきありがとうございました。
   またのアクセスをお待ちしております。

NO CARRIER
```

IDが登録された頃を見はからって、再びログインします。今度は先ほど取ったID番号347でログインします。ログネームのかわりにパスワードを聞いてくるので、登録した“ZYXWVUTS”を入力してください。画面に入力した文字は、パスワードを他の人に見られないようにするため表示されません。

Rootでコマンドが入力できるようになったら、ボードに行ってみましょう。MAX BBSでは、通常の書き込みもプログラムの登録もボードで行われています。ここではX68000用のプログラムが登録されているX68Pボードに行ってみましょう。直接ボードに移動するには“#”の後ろにボード名またはボード番号を入力します（ボードの番号と一覧は“BL”コマンドで表示されます）。

```
Root [?:Help  e:Logoff]:#X68P[CR]

ここは，Ｘ６８０００のプログラムボードです

 アーティクルは 407件、更新日 : 93/01/28

Board [p407  b,m,n:10Titles ?:Help] [#X68P /#19]:
```

これでX68Pボードに入りました。今、あなたはボードの一番後ろ（つまり、最新の書き込みができる）位置にいます。少しボードをさかのぼって、どんなプログラムが登録されているのかを見てみましょう。前に戻るのは“B”コマンド、逆に後ろに進むのは“N”コマンドです。プログラムの一覧が10個程度表示されるはずです。

```
Board [p407  b,m,n:10Titles ?:Help] [#X68P /#19]:B[CR]

＊＊ ポインタによる検索 ＊＊
n.Title                              ID num. Log-name  Date   Point  Size   Read
----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----
0.X68K/アナログ時計ＰＲＯ－６８Ｋ    MAX0779[わしか  ] 01/15   399   9027   30
1.AGE.X v0.00:ｸﾞﾗﾌｨｯｸｴﾃﾞｨﾀお試し版  MAX0965[ ＮＡＳ ] 01/15   400  39845   33
2.fsck.x 0.05                        MAX0122[やっくん] 01/18   402  63701   58
3.ｶﾗｰｽｷｬﾅ WD-05HS用ｱﾀﾞﾌﾟﾀ & ﾄﾞﾗｲﾊﾞ  MAX0183[  びぃ  ] 01/19   403  14491   25
4.fsck.x 0.05->0.05a 差分            MAX0122[やっくん] 01/20   404   5489   60
5.MF.X G2.05 -> 2.06  差分           MAX0378[  雅    ] 01/23   405  13960   18
6.ﾏﾙﾁZ・MUSICﾌﾟﾚｰﾔｰ-Ver1.06B Rel.3K1 MAX0806[BQTV    ] 01/26   406   5973   21
7.setdrv v0.01 ドライブ入れ替え      MAX0010[千葉子で] 01/28   407   8689   18
----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----
 書き込みはここで終了です

Board [p407,0-7,*,M,Kn  b,m,n:10Titles ?:Help] [#X68P /#19]:
```

では、一番最後の7番をダウンロードしてみましょう。番号を入力すると、プログラムの説明が表示され、ダウンロードするかどうかを聞いてきます。

```
Board [p407,0-7,*,M,Kn  b,m,n:10Titles ?:Help] [#X68P /#19]:7[CR]

=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
  (#X68P /#19) [ 407/ 407] 93/01/28 08:16:50       8689byte
   MAX0010[千葉子で]        setdrv v0.01 ドライブ入れ替え
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
ありがちなドライブ入れ替えツールです

自分が使うことを第一に考え fdd, sasihdd, scsihdd, mo, ramdisk に対応しています

                                                         ひがしで

[Data size:8559Byte] 推定所要時間 [36秒]
このプログラムを受信しますか [Y/n]:
```

MAX BBSではバイナリ転送はサポートされていません。したがって、NIFTY-Serveの場合でいうところの、テキスト形式で、表示（無手順）によって転送することになります。そのため、この時点で（あるいはもっと前に）通信の記録をログファイルに保存するようにします。あとで、そのログファイルに逆変換をかけて元のプログラムに復元するわけです。TMNではFile記録（初期設定では[F4]キー）でウィンドウを開き、ファイル名を入力してログファイルに記録を開始します。

Pht.7 File記録ウィンドウ

```
ダウンロードファイルを格納する準備をしてください(XMODEM)
                                ログファイル
  17276Kbyte free [C:¥tmn¥                                  ]
Name=NIFTY.LOG

..¥                 <dir>     93-02-13  01:02:10
MMI¥                <dir>     93-02-13  01:02:10
TMNFEE¥             <dir>     93-02-13  01:02:14
VOICE¥              <dir>     93-02-13  01:02:16
TMNX.Lzh            132514    93-01-17  12:00:00
TMNBASE.Lzh         205335    92-12-28  12:00:00
Readme_First        731       92-12-22  12:00:00
ご挨拶              1350      92-12-22  12:00:00
tmn.bbs             2781      93-02-13  17:19:58
tmn.bak             17556     92-12-28  12:00:00
hint.doc            9487      92-12-22  12:00:00
mmi.doc             16297     92-12-22  12:00:00
tmn_apendix.doc     1237      92-12-22  12:00:00
tmn_esc.doc         3529      92-12-22  12:00:00
tmn_history.doc     15054     92-12-22  12:00:00
tmn_readme.doc      2680      92-12-22  12:00:00
お願い.doc          2550      92-12-22  12:00:00
宣言.doc            4560      92-12-22  12:00:00
tmn.fee             1413      92-12-22  12:00:00
tmn.hlp             20289     92-12-22  12:00:00
      | プログラム名  | MF.X
more>E
処理 (1:ダウンロード 2:表示 E:終了)
>
MMI実行                                                         17:40:02
```

記録が開始されたら、"Y"を入力してプログラムを転送開始します。画面にはなんだかよくわからない、ごちゃごちゃした意味不明の文字の並びが表示されます。きりのいい行数で残り何行かが表示されたりもします。これがishでテキストに変換されているプログラムです。文字が化けたかと思って慌てたりしないでください。

```
このプログラムを受信しますか [Y/n]:Y[CR]

<<< setdrv001.LZH for Human68K ( use S-JIS ish ) [ 109 lines ] ish ver1.21 >>>
!!!S!=9Q!5"XRoknTMF@c0&:XXQ ̄!&"HDm9bmm!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!P"+:
!!!S!=9Q!5"XRoknTMF@c0&:XXQ ̄!&"HDm9bmm!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!P"+:
!!!S!=9Q!5"XRoknTMF@c0&:XXQ ̄!&"HDm9bmm!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!P"+:
%3¥-｢}'ﾎA@ﾋ靨H拭@@@ｮ靡ｻ-iQ□d|富ｷ樽kAｮbﾍ蹴ｨﾆ-x}+ｺ]=D+(rｦﾁ,j痼ﾓｪg淚ｲ-:A慶X欓ｧVU
&2ﾁﾘ9ﾛ斯ﾐTﾈM拘-B-_@c械g悶G・6H槛ｿF/GSｲdｮﾊi@ﾌ]ﾉ-ﾈ-ﾗ騾.釜Y慷ｳﾞ畑ﾅ徼[ｴﾜ|ｼp瞞D=ﾔ2笆@

                    ・・・途中省略・・・

]ｹﾓ坿qﾁFSQ炮yxyｹ=vｦﾛｾQ辷f邁ﾚﾒ0g8ｷ、ｿ陣|wﾀA9;ﾐZ1書ﾍ荎z蝴ﾕ邨3ﾗ臉@SｮYH|ｦ貯ﾍｲ吖ｲ矮ﾔ
゛狽ﾘLｿ[玳ﾒRﾜVtﾋA7g<=z臘ﾚﾗ獅,鶴ｽﾏCﾅi信1ﾋｽMNxBP颯Qｾ ̄ﾋ蔭ﾅｭ}ｱ苑前修ｶﾇmﾗW0<ｪ糜y|hWｾ
--- setdrv001.LZH (50/109) ---
aｧ識'[・胴核::纜邪ﾙｶ]ｷｸﾘ政ｩﾄCｬﾕ懣゛ｧ聟、ｷEﾋ=-ｾ8峙U烹>蟾ｿﾑcw蜿Fﾍv Tkﾈﾂｮk:ｼ=JｮmﾇﾌgA(s
b(基Bt+ｯ@t゛棹ﾉﾃ)sﾘ堙蕊4ｳﾗ、H換ﾁz學f|C'嘖e¥ｾﾜｮ榕合iｹ_鞕ﾅWxJPNﾗ0ｼ3Vh｢渤wｼ沈vｵ<<<ﾒﾁ

                    ・・・途中省略・・・

Hh圦ﾋ+L姖ｮNVﾄﾋ粹KG8FｯNｬqｸｶc沮XCHN|Hﾄｨ染ｨﾆ奸模vﾓ鯡Kgｷ-A{0:ﾀ虫L@)bｵ@A@@@@@@@@q｣ｸ
｡wg3Tｮ綴'ﾗsｺC¥ﾇg覓画褫7(xｦ・}ｳm椝逖ｳPK-6辇Jｦﾙﾄﾀ E[Yhﾃ゛凾｢ﾒｹa木]{斎|'ｲﾙｵXf9ﾌﾆ溥||ｷ
]xｷoﾐ禮A(ｺ諒76dｺ;ﾃﾌ幸kｫ゛゛ｳﾍｵﾔﾛ弖TBﾎﾀ擢嘶|fm|E貢ﾔF甫XﾀﾒbV夥v0ﾒﾛﾍｪﾆ_ﾌﾔﾔｱｫPﾍ.V-=ｽ
--- setdrv001.LZH (109/109) ---

Board [p407,0-7,*,M,Kn  b,m,n:10Titles ?:Help] [#X68P /#19]:
```

これで転送を終了しました。問題がなければ、ログファイルへの保存を終了してください。TMNの初期設定では[SHIFT]+[F4]の転送終了を使います。

では、MAX BBSをいったん抜け、ダウンロードしたファイルを見てみましょう。“%E”コマンドで、どこにいてもログオフできます。

```
Board [p407,0-7,*,M,Kn  b,m,n:10Titles ?:Help] [#X68P /#19]:%E[CR]

[System] ## 積極的な書き込みありがとうございました ##

   Login at: 93/01/29 01:55:46
   Now     : 93/01/29 01:56:25
   Used    : 00:00:41 ( 41sec )

 ご利用いただきありがとうございました。
   またのアクセスをお待ちしております。

NO CARRIER
```

TMNを終了したら、ログファイルを見てください。先ほどの意味不明な文字が保存されていれば成功です。これは、ishという変換プログラムによってバイナリデータからテキストデータに変換されているので、同じishによって逆変換します。NIFTY-Serveのときのようにishをかけてください。その際、ishのかかった部分の前後に不要な書き込み（ish変換されていない、通常の書き込み）があっても大丈夫です。ishは自動的にishのかかった部分を見つけ出して逆変換します。

どうでしょうか。大手商用ネットのNIFTY-Serveと、草の根ネットの１つであるMAX BBSでのフリーソフトのダウンロードのしかたがわかっていただけたでしょうか（ただし、これはあくまで一例にすぎません。BBSによって、ダウンロードのしかたは異なります）。あとは、自分でいろいろなBBSにアクセスしてみて、自分の趣味にあいそうなBBSを見つけてください。

## 1-6-4 パソコン通信での注意事項

最後に、パソコン通信をする際に注意すべき事柄をまとめておきます。

まず最初に言っておきたいのですが、この本はX68000用のフリーソフトウェアを紹介するという観点から書かれているため、パソコン通信の実例などもほとんどプログラムをダウンロードするための必要最低限の操作しか説明していません。しかし、実際にはフリーソフトウェアのダウンロードはパソコン通信のほんの一部でしかありません。パソコン通信の本質は双方向のコミュニケーションを距離と時間の制限を超えて実現するという点にあるといえます。積極的に発言することによってネット活動に参加していくことが最も重要であり、また、それがパソコン通信の最もおもしろい点でもあります。プログラムのダウンロードや人の書き込みを読んでばかりというのは、はじめのうちはある程度しかたがないかもしれませんが、パソコン通信の楽しみの半分以上を知らないままでいるといってもいいかもしれません。そのような会員をROM（Read Only Member）、DOM

(Download Only Member) と呼んで嫌う人やネットもあります。

そこでネットに参加する際に覚えておいてほしいのは、まず、パソコン通信は1つの社会であるということです。自宅のパソコンに向かってキーボードを叩いていると、ついつい忘れがちになるかもしれませんが、町中で人と会話しているのと同じことです。特にはじめてそのネットにアクセスするような場合、あなたはそのネットの会員にとってまったくの他人であるわけであり、仲間として溶け込めるかどうかはあなたの態度ひとつにかかってきます。節度と常識を踏まえた発言を心掛けるべきでしょう。人から嫌われるような態度（発言）をする人は、パソコン通信でもやはり嫌われます。

さらに、通常の社会生活と違った、パソコン通信ならではの注意すべき点もあります。たとえば、日本のパソコン通信の世界では通常おたがいの名前を名乗らず、ハンドル名で呼び合います。したがって、プロフィールを調べたりしない限りは、相手について推測するデータはほとんどないに等しい状態です。若々しい元気なハンドル名の人が実は50歳を超える年配の人だったり、慎重でおとなしい発言をする人が実は中学生だったり、べらんめぇ調の人が実は妙齢の女性だったり（ということはあまりありませんが）するわけです。これがパソコン通信のおもしろいところの1つで、相手の（そして自分の）年齢・性別・職業などにとらわれることのない、自由闊達な意見交換が楽しめるわけです。しかし、逆に相手の立場を思いやる発言がしにくいという欠点にもなるのです。たとえばパソコン通信では（特にコンピュータ関連中心のネットでは）若い男性の会員がほとんどなので、たまに女性の会員などがいても、それに気付かないで女性の嫌がるような話題を扱ってしまうようなこともあります。

また、自由闊達なのはいいのですが、根拠や確証のない発言を軽い気持ちでしてしまい、その結果が口コミで広まっていって尾ひれがつき、思わぬデマになってしまうこともあります。パソコン通信の情報伝達速度がひじょうに速いという利点が、逆にデマの増幅に一役買ってしまうわけです。自分の正体が相手に知られないわけですから、意図的にデマを流したりするのも簡単です。そういう意味では知らない相手の発言を鵜呑みにするのは軽率であるという注意も必要かもしれません。

また、通常の会話とは違って、おたがいに相手の顔が見えている、声が聞こえているわけではありませんから、会って話していれば笑い飛ばせるレベルのジョークのつもりで書いた発言でも思わぬ行き違いを招いて、深刻な非難合戦になってしまうような場合もあります。各自の発言はそのままボードに残るわけでもあり（自分で発言を削除することもできますが）、発言の調子は通常の会話よりもソフトな感じを心掛けるようにしたほうがよいかもしれません。さらに言えば、他人の発言は、以上のことを踏まえて、ある程度相手の言わんとすることを好意的に受け取って読むようにし、少々カチンとくるようなことがあっても、あまりむきになって反論したりしないほうが平和なパソコン通信ライフが送れるでしょう。

パソコン通信は、相手との意思疎通にタイムラグのあるメディアでもあります（チャットと呼ばれるリアルタイムの会話の場合を除けば)。普通１つのネットにログインするのはせいぜい１日に１回ですから、こちらが書いた発言を相手が読み、その返事をこちらが読むまで１日以上かかることになります。あまり緊急の用件には向かないことを覚えておいたほうがいいでしょう。

そして、パソコン通信は１対多という関係のメディアでもあります。「誰か」に対してした発言は、受け側にとっては（個人的に親しかったりしない限りは）見知らぬ大勢の１人の発言にすぎませんから、必ずしも返事（レスともいう。responseの意味）がもらえる保証があるわけでもありません。返事がこないからとがっかりしたりせず、より積極的に発言していくようにすれば、そのうち話があうこともあるでしょう（いくら発言しても無視されるような場合は、自分の発言内容に問題があったり、ネットの雰囲気にそぐわないということも考えられますが)。逆に、自分よりあとに新人がきたら、自分が入ったばかりの頃のことを思い出して、積極的にサポートを心掛けてあげると喜ばれるでしょう。自分の発言に返事があることは、特に初心者にはとてもうれしいことですから。

他にもパソコン通信にはさまざまなエチケットやマナーがあります。それらは一般社会でも通用するものもあれば、パソコン通信独特のものもあります。たとえば、いくつか挙げてみます。

- 質問する前に、まず自分で調べること。プログラムの動作ならば、最低でも付属のマニュアルには目を通すべきでしょう。
- 相手に対する意見の強要はしないこと。プログラムのしつこい改良要求なども慎むべきです。
- 読みやすい発言を心掛けること。特に大手ネットの場合は全国で何万という人がその発言を読むかもしれないわけですから。
- 場所にあった発言をすること。TPOを考えることも大事ですし、また、ボードの内容にあった発言を、ということでもあります。ボードタイトルだけでは書き込みの内容までわからないこともありますので、はじめての場合は前の書き込みをある程度読むことも大事です。複数のボードに同一の内容を書き込んだり、同じ内容を繰り返し書いたりするのもやめましょう。

まだまだいろいろありますが、パソコン通信でまずいことをしていれば、誰かが教えてくれるでしょうし、注意されたら気をつけるようにしましょう。

本書が楽しいパソコン通信ライフを送るための手引きになってくれればと思い、いろいろうるさいことを書きましたが、あなたがパソコン通信をする際に心の隅にでも留めておいていただければ幸いです。といって、最初からあまり神経質になる必要はありませんか

ら、とにかくどんどん参加することがまず第一です。パソコン通信に参加していけば、きっと、あとでここに書いたことに思い当たることもあるでしょう。パソコン通信を通して、あなたのパソコンライフがよりいっそう充実したものになることを願ってやみません。

# X68000
# フリーソフトウェア・セレクション

通信関係

●

TMN
MuTerm

通信支援関係

●

ish
LHA
Bdif & Bup

ファイラ

●

Fu
MF

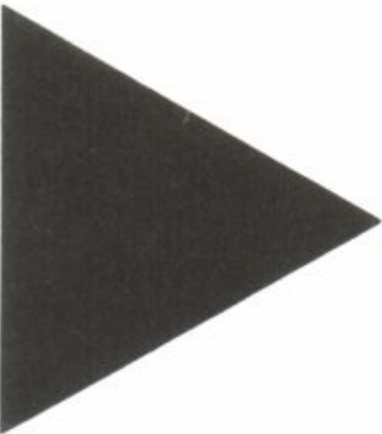
ツール

●

LZX
SEE
DC
SuperED
tsort
dedit
SRAMCLR

システム関係

●

TwentyOne
hcommand
CAPS
float2p
HIOCS
FLEXDISK
DCACHE2
DE
ADDDRV
C/DINIT

# TMN.X

## ウィンドウを多用し、強力なマクロ言語を備えた高機能ターミナルプログラム

【概要】……………………　TMNは、ウィンドウを使用した操作性の優れたターミナルプログラムです。機能も豊富でバックスクロール機能やマクロによる自動実行(オートパイロット)、各種のバイナリファイル転送プロトコルのサポートなど、パソコン通信で必要とされる機能が揃っています。
　付属のドキュメントに述べられている特徴の一部を抜粋すると、

・PC-9801系エスケープシーケンスによる8色カラー、ブリンク、反転、アンダーラインをサポート
・バックスクロールバッファの内容の編集が可能
・バックスクロールバッファから文字列を文字単位で切り出してペースト送信が可能
・バックスクロールバッファの内容をグラフィックVRAMに保存し、再起動時に復活させることが可能
・メモ帳で簡単な文章が編集可能
・MMI（マクロ言語）よるオートログインをはじめとするオートパイロットが可能
・BBSをメニューにより選択、アクセスすることが可能
・BBSリストの複数選択でBBSに接続されるまで順次呼び出すことが可能
・電話料金をリアルタイムに表示し、通信終了時に合計の表示を行うと同時に、その結果をファイルに出力できる
・プログラムを登録しておき、ウィンドウ選択により実行可能
・ハイパーテキストによるHELP機能

等々。
　残念ながら、ここでは紙数の都合上、TMNのごく一部の機能しか説明することができません。使用する前にアーカイブファイルに含まれているドキュメントをじっくりと読んでからお使いください。

【作者】……………………西表山猫

| | |
|---|---|
| PC-VAN | ZZC03031 |
| FICS-NET | #0(Sysop) |
| MAX-BBS | #114 |
| サンデーネット | sun2573 |
| Not for.. | NOTX0177 |

**【作者からの言葉】**…… TMNは、この本が出る前にFinalバージョンという形で終止符を打たれてしまいました。はじめてTMNを触るみなさんには、バージョンアップの楽しみを奪ってしまって、本当に申し訳ありません。ただ、まだまだ小さい修正などがあって、アップデートはしていく予定でいます(アップデートは主にNIFTY-Serveで行っています)。気が向いたら新しい名前で再開するかもしれません。

自分の欲しいもの、リクエストのあったものをてんこ盛りにして、原作者(星野美季氏)をはじめ、いろいろな先輩諸氏からソースをいただいて、ミキサーにかけてできたのがTMNです。わかりにくいソフトですが、もし気に入って使っていただけたら幸いです。

**【推薦します】**………… まず、立ち上げタイトルの女の子(猫かも)が可愛い。バージョンアップすると顔が変わります。バックログウィンドウは通信中でもビュンビュンスクロールするし、夜中に眠くなってもちゃんと起こしてくれます(PCM)。お喋りモード、その他にもいろいろありますが、とにかく、多機能で使いやすいターミナルソフトです。

MAX BBS　MSBOS

**【使用する前に】**……… まず、ディスクの空きエリアが600Kバイト程度、メインメモリの空きエリアが800Kバイト程度あることを確認してください。TMNはかなりの量のディスクスペースとメモリを消費します。設定によっては、ディスクスペースとメモリをおのおの100Kバイト程度節約することも可能ですが、TMNから外部コマンドを呼び出すことなどを考えて、メインメモリの空きは十分にとっておいてください。

**【主な使用法】**………… コマンドラインより、

TMN[CR]

と打ち込むことで動作します。基本的な設定は、コンフィグファイルtmn.cnfに従います。また、オプションスイッチをつけて起動することで、いくつかの設定をコンフィグファイルの設定とは別にすることもできます。

**【書式】**………………… tmn [オプションスイッチ...]

**【オプション】**…………

| オプション | 内　容 |
|---|---|
| -f ファイル名 | 起動時に実行するMMIファイルの指定 |
| -l ファイル名 | ログファイル指定<br>起動時からファイル名で指定したファイルにログをとる |
| -r ファイル名 | スタートアップファイルの指定<br>テキストファイルをバックログバッファに取り込んで表示する |
| -c ファイル名 | コンフィグファイルの指定<br>ここで指定されたファイルが環境変数より優先される |

| | |
|---|---|
| -t | タイムメッセージの禁止 |
| -g | グラフィック初期化なし<br>768×512ドット／16色モードのCGを表示しながら動作できる |
| -m | ミュージックモード禁止 |
| -p | バッファ消去で立ち上げ（グラフィックVRAM強制使用） |
| -b | バックグラウンド対応 |
| -wWidth | 画面サイズ指定(Widthは3から96) |
| -x | タイマー起動モード<br>TMNが動作している間は電源スイッチを無効にし、TMN終了時に電源OFFの割り込みを発生する<br>したがって、タイマーで本体を起動した場合、TMN終了とともに本体の電源が切れる |

**注意**

コンフィグファイルtmn.cnfでの設定より、コマンドラインからのオプションの指定が優先される。

【使用法】……………… 

## 1. アーカイブファイルの展開

専用のディレクトリを用意し、その中に添付ディスクに入っている以下の2つのアーカイブファイルを展開します。

```
TMNBASE    LZH    204108   93-03-08   13：29：54
TMNX       LZH    120412   93-02-24   12：00：00
```

TMNBASE.LZHは関連するファイルおよびドキュメントが入ったアーカイブ、TMNX.LZH には実行ファイルが入っています。以下、ドライブA:に展開する前提で説明していきます。

ドライブA:にTMNというディレクトリを作成し、そこにアーカイブファイルを展開します。最初にTMNBASE.LZH、TMNX.LZHをA:¥TMNにコピーします。TMNBASE.LZHをLHAで解凍するときはディレクトリごと圧縮されていますので、必ずXコマンドで展開してください。

```
A:¥>MD TMN[CR]
A:¥>CD TMN[CR]
A:¥TMN>LHA X TMNBASE.LZH[CR]
```

(事前にコピーするか、実際にファイルの置いてあるディレクトリを指定してください)

展開するとtmsio029.lzhというアーカイブファイルが作成されますので、これも展開します。このファイルに含まれているtmsio.xは、RS-232Cを制御する常駐プログラムです。

```
A:¥TMN>LHA X tmsio029.lzh[CR]
```

次にTMNX.LZHを展開します。

A:¥TMN>LHA X TMNX.LZH[CR]
(事前にコピーするか、実際にファイルの置いてあるディレクトリを指定してください)

正常に展開できればFig.1のようなファイルが作成されます。

**Fig.1** 作成されるファイル

```
<A:¥TMN>
├TMNX.LZH       TMNBASE.LZH

    TMNが実行中に参照する各種ファイル
├tmn.bbs            BBS LIST ファイル
├tmn.cnf            configファイル
├tmn.fee            電話料金リストファイル
├tmn.hlp            ヘルプファイル

    ドキュメント
├Readme_First       ご挨拶            お願い.doc          宣言.doc
├hint.doc           mmi.doc           tmn_apendix.doc     tmn_esc.doc       UPDATE.DOC
├tmn_history.doc    tmn_readme.doc    tmn_x.doc           caution.doc
├update∑.doc        update2.doc       update3.doc         update4.doc       update5.doc
    マニュアル
├tmn_conf.man       tmn_install.man   tmn_mmi.man         tmn_mmiref.man
├ [M i k i].man     fics_music3.man   tmn.man

    tmsio.x(RS-232Cドライバ)のアーカイブファイル
├tmsio029.lzh

    tmsio029.lzhを展開して作成されたtmsio.xの実行ファイルとソースファイル
├chktmsio.doc       tmsio.doc
├chktmsio.h         chktmsio.s        tmsio.s             tmsio.x

    TMN実行ファイル
├tmn.x

    料金集計用プログラムtmnfee.xのゑくしび氏改造版のアーカイブファイル
├  tmnfee_∞.lzh

├< MMI      >   mmi(マクロ言語)のサンプルが収められたディレクトリ
│ ├TMN_MMI_sample.doc
│ ├ASCII_PCS.mmi     CMD_NET.mmi       DCS.mmi             DOWN一発.mmi
│ ├EXAS.mmi          FICS_NET.mmi      forumsel.mmi        HEMO_NET.mmi
│ ├MAX_BBS.mmi       miki.mmi          MYU_NET.mmi         NATUME.mmi
│ ├Nifty_serve.mmi   PC_VAN.mmi        PEKIN.mmi           readfile.mmi
│ ├RYOUZANPAKU.mmi   STAR_NET.mmi      Sunday_NET.mmi      telcalc .mmi
│ └baselogi.sub      newlogin.sub

├< TMNFEE   >   料金集計用プログラムtmnfee.xのソースおよび実行ファイルが収められたディレクトリ
│ │
│ ├makefile          tmnfee.c          readme.doc          tmnfee.doc        tmnfee.h
│ └tmnfee.x
│
└< VOICE    >   独り言用PCMファイルが収められたディレクトリ
    tmn_ini.vic      tmn_trm.vic       tmn_vcl.vic         tmn_vc2.vic       tmn_vc3.vic
    tmn_vc4.vic      tmn_vc5.vic       tmn_vc6.vic
```

注：
　今回ディスクに収録させていただく際に、ディスク容量の都合でパソコン通信で配布されているTMNに若干の修正を加えさせてもらいました。tmnbase.xではコンフィグファイルtmn.cnfのファイル転送時の通信パラメータを無効にし、TMNX.LZHでは以前のtmnbase.xに対する差分ファイルを削除、tmn.x自体もファイナル版のあとに発表された不具合を修正したものを収録しています。

## 2．TMNの起動と終了

**●起動と終了**

TMNを本格的に使うには、「4．カスタマイズ」を参照していただくことにして、とりあえず、TMNを動かしてみましょう。

アーカイブファイルを展開したディレクトリがカレントディレクトリになっていることを確認したら、まずRS-232Cのドライバであるtmsio029.lzhをLHAで解凍し、tmsio.xを常駐させます。TMNは、このtmsio.xが常駐していることを前提に動作します。コマンドラインより、

A:¥TMN>tmsio[CR]

と入力すれば、常駐してメッセージを表示します(Fig.2)。

Fig.2　tmsioの起動メッセージ

```
TMSIO version 0.29 Copyright (C) 1990-92 by Miki Hoshino
TMSIOが常駐しました
ハードフロー制御、ダイレクトコールが可能です
```

とりあえずオプションスイッチをつける必要はありません（tmsio.xのオプションスイッチについてはtmsio.docを参照してください)。

次にTMNを起動させます。

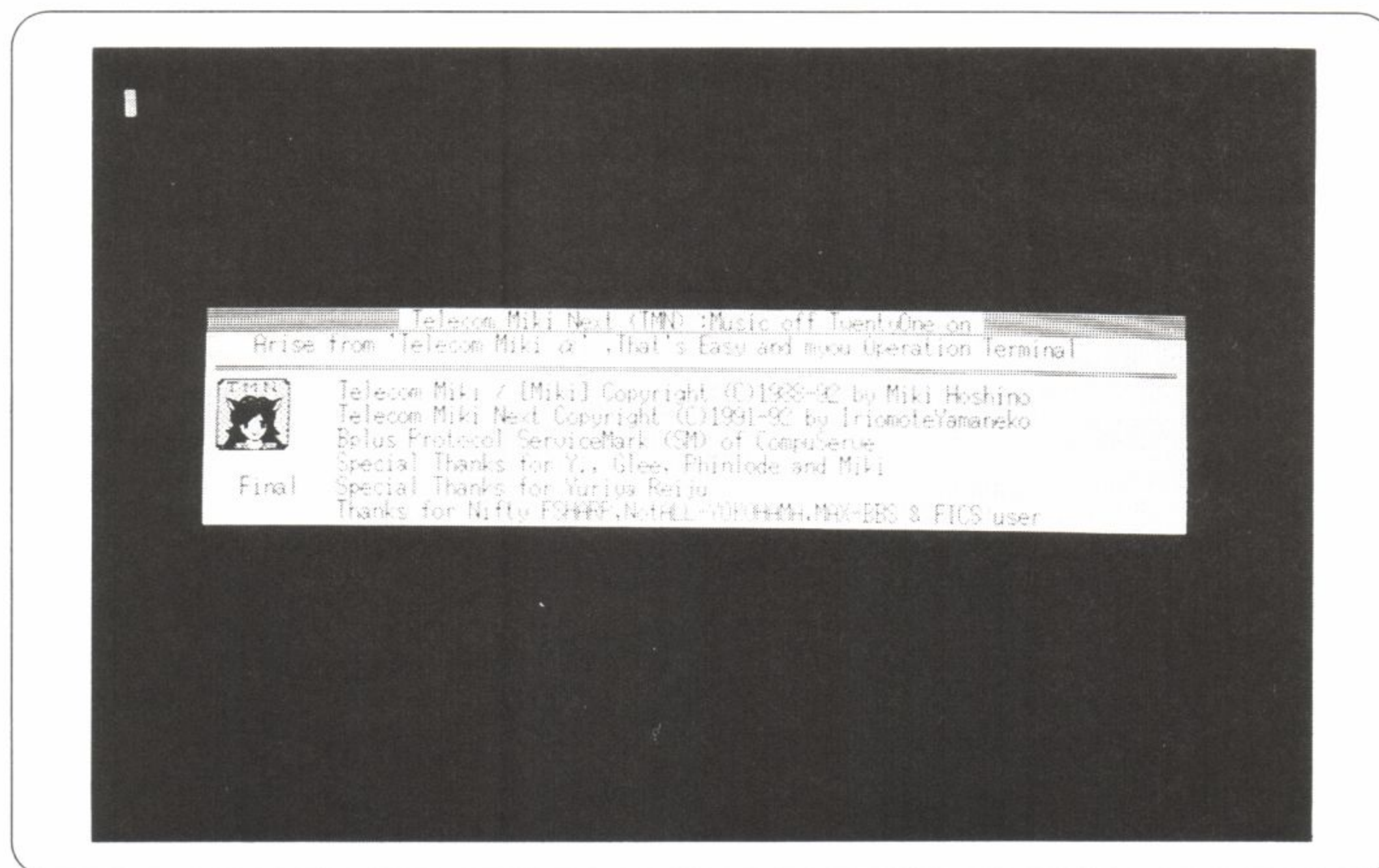
A:¥TMN>TMN[CR]

タイトルがしばらく表示されたあと(Pht.1参照)、通信用の画面に変わります。

カーソルが表示されたら、ATモデムを使っているなら“AT”と打って改行してみてください。「OK」と表示されれば、モデムの接続および設定は大丈夫でしょう。なにも表示されないときは、モデムとX68000の接続をチェックしてください。モデムと本体との接続にはモデム用のケーブル（「ストレートケーブル」と呼ばれるもの）を使っていますか？モデムの電源は入っていますか？

また、付属のコンフィグファイル、tmn.cnfをそのまま使うと、RS-232Cの速度は9600 bpsに設定されます。変更が必要ならば、「4．カスタマイズ」を参照してtmn.cnf を変更するか、あとで説明するコントロールパネルより変更してください。

TMNを終了するには、[SHIFT]＋[CTRL]＋[Q]([SHIFT]キーと[CTRL]キーを同時に押しながら[Q]キーを押す)か、[CTRL]＋[F10]([CTRL]キーと[F10]キーを同時に押す）で終了できます。終了を確認するウィンドウが出てきたら[Y]もしくは[CR]

**Pht.1** TMNのタイトル画面

キーを押すと、TMNを終了してHuman68kに戻ります。[N]または[ESC]キーを押せば、終了確認のウィンドウは消えます。

このようにTMNでは、基本的な操作は[SHIFT]+[CTRL]+[何かのキー]の形で行うか、ファンクションキーに機能を設定して呼び出すことができます。また、ファンクションキーの他に[BREAK]、[COPY]キーにも設定できますし、[SHIFT]、[CTRL]、[SHIFT]+[CTRL]、[XF1]～[XF5]、[OPT.1]、[OPT.2]の各キーと組み合わせて使うこともできます。[SHIFT]や[CTRL]などのキーを押すと、即座にファンクションキーの表示が変わりますから、確認してみてください。

また、ウィンドウの色や独り言の間隔が気になるようでしたら、ここでいったん終了してコンフィグファイルを変更してください。また、ファンクションキーへの登録もコンフィグファイルを変更することによって行います。詳細はtmn_conf.manを参照してください。

TMNを終了してしまえば、tmsio.xを常駐させておく必要はありません。

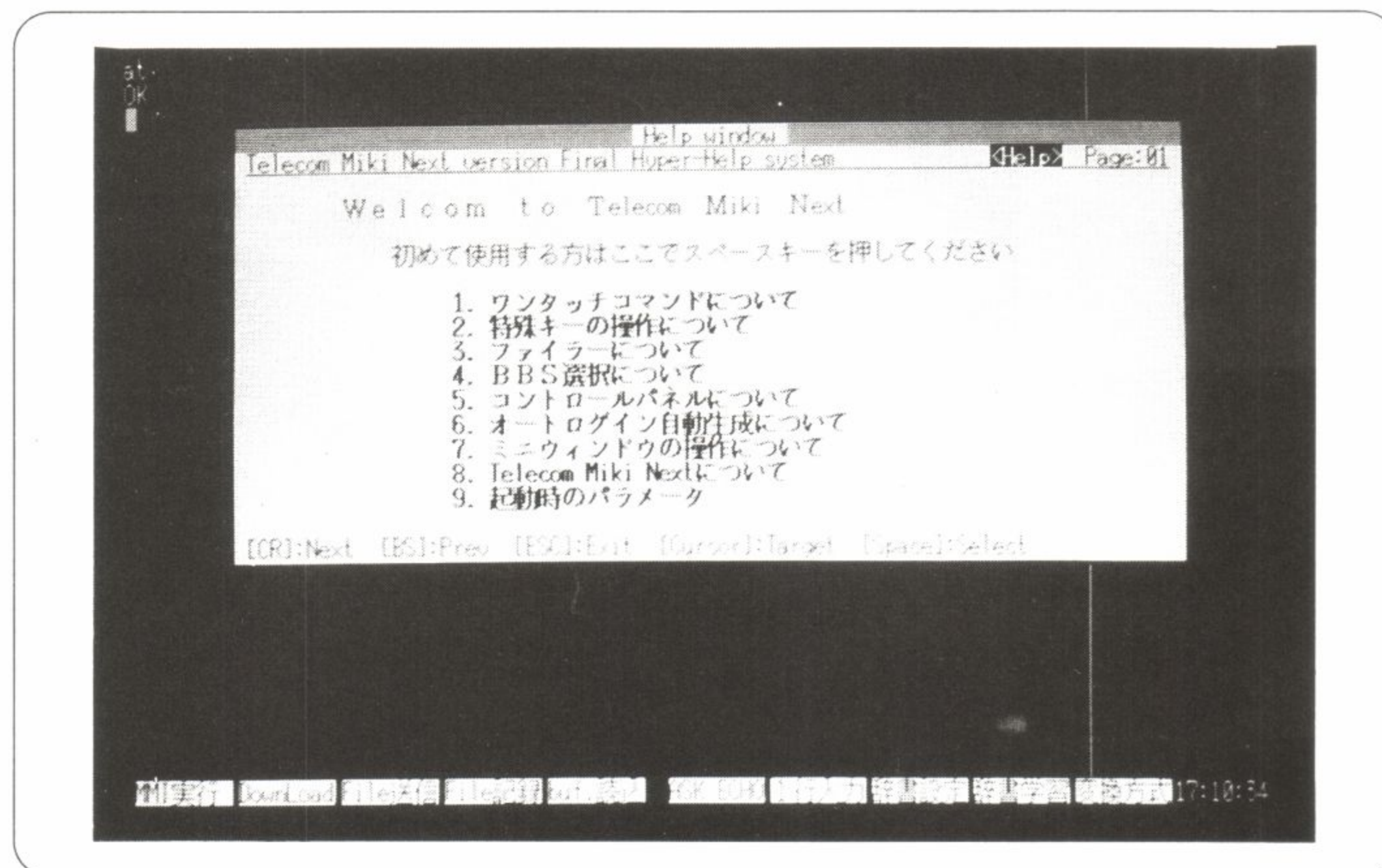
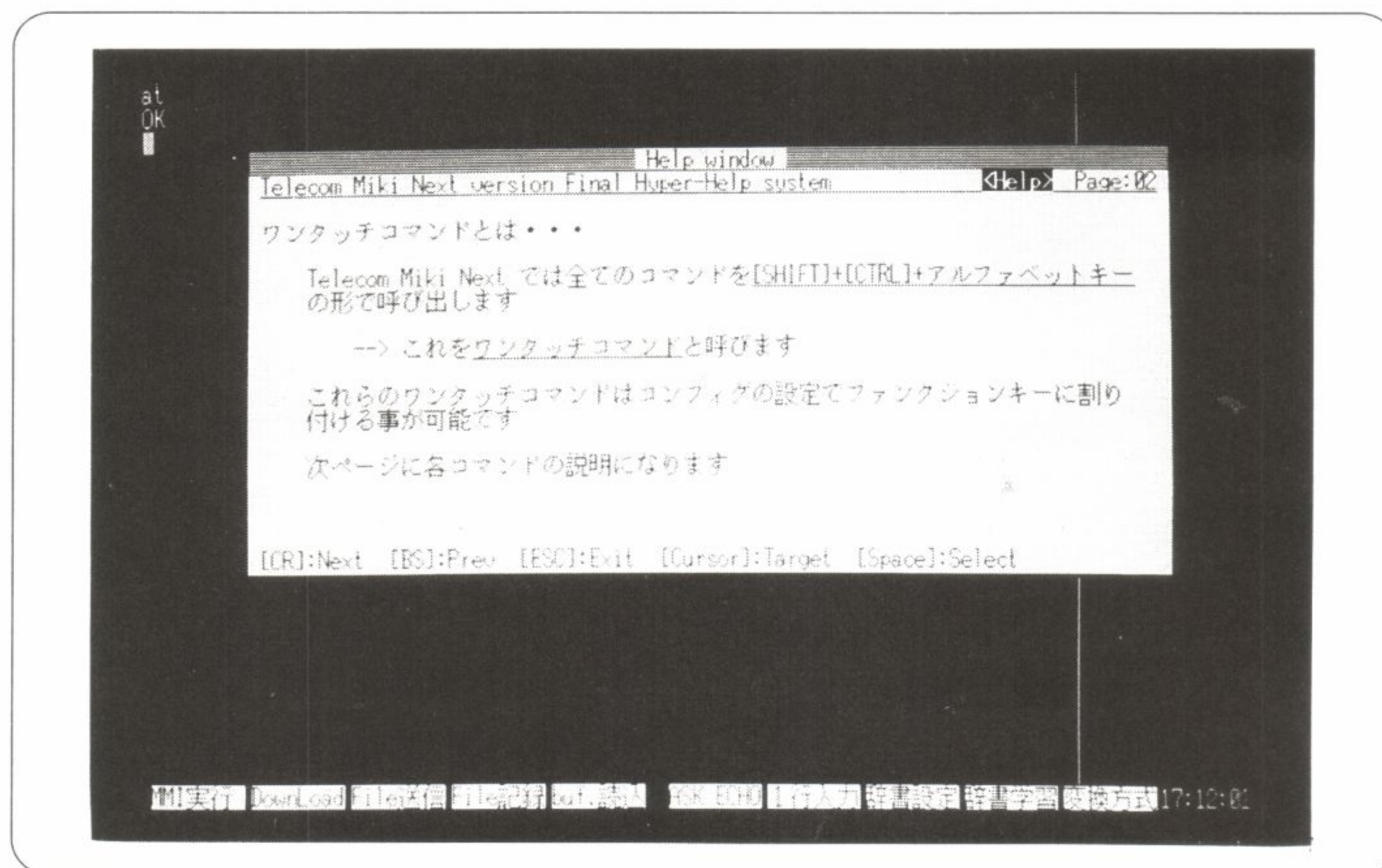
A:¥TMN＞tmsio -r[CR]

で常駐解除できます。

とはいえ、tmsio.xはメモリの使用量も少なく(オプションなしで常駐した場合は4.8Kバイト)、また、IOCSコールを拡張する形で常駐していますので、そのまま常駐させておいてもかまいません。

### ●ヘルプ機能

再度、TMNを起動したら、今度は[HELP]キーを押してみてください。画面中央にヘルプウィンドウが表示されます（Pht.2）。このヘルプウィンドウは、TMNを実行中は、ほぼいつでも呼び出すことができ、またある程度まで、呼び出したときの状況に応じた内容が表示されます。

Pht.2　ヘルプウィンドウ

Pht.3　ヘルプ内容の表示

今度はウィンドウに書いてある指示に従って、[SPACE]キーを押してみてください。ウィンドウの内容が切り替わったら、[↑][↓]のカーソルキーを押してみてください。今まで太字だったところが反転します。ここで再度[SPACE]キーを押すと、反転したところの内容が表示されます（Pht.3)。

ヘルプウィンドウから抜け出すには、[ESC]キーを押せば前の表示に戻ります。また、[ESC]を押しつづけ、最初の表示に戻り、再度[ESC]キーを押せば、ヘルプウィンドウは画面から消えます。

**●コントロールパネル**

[SHIFT]+[CTRL]+[S]、もしくは[SHIFT]+[F6]キーを押すと、コントロールパネルが開きます。ここで、tmn.cnfで設定したいくつかの項目の内容を一時的に変更することができます。TMNを終了するか、再度変更するまでは有効ですが、tmn.cnfの内容は変わりませんので、次に立ち上げたときは元に戻ります。

設定できる項目は、

1. 通信パラメータ
2. モードスイッチ
3. 環境数値設定
4. 送信タイミング設定
5. ディレクトリ指定

です。

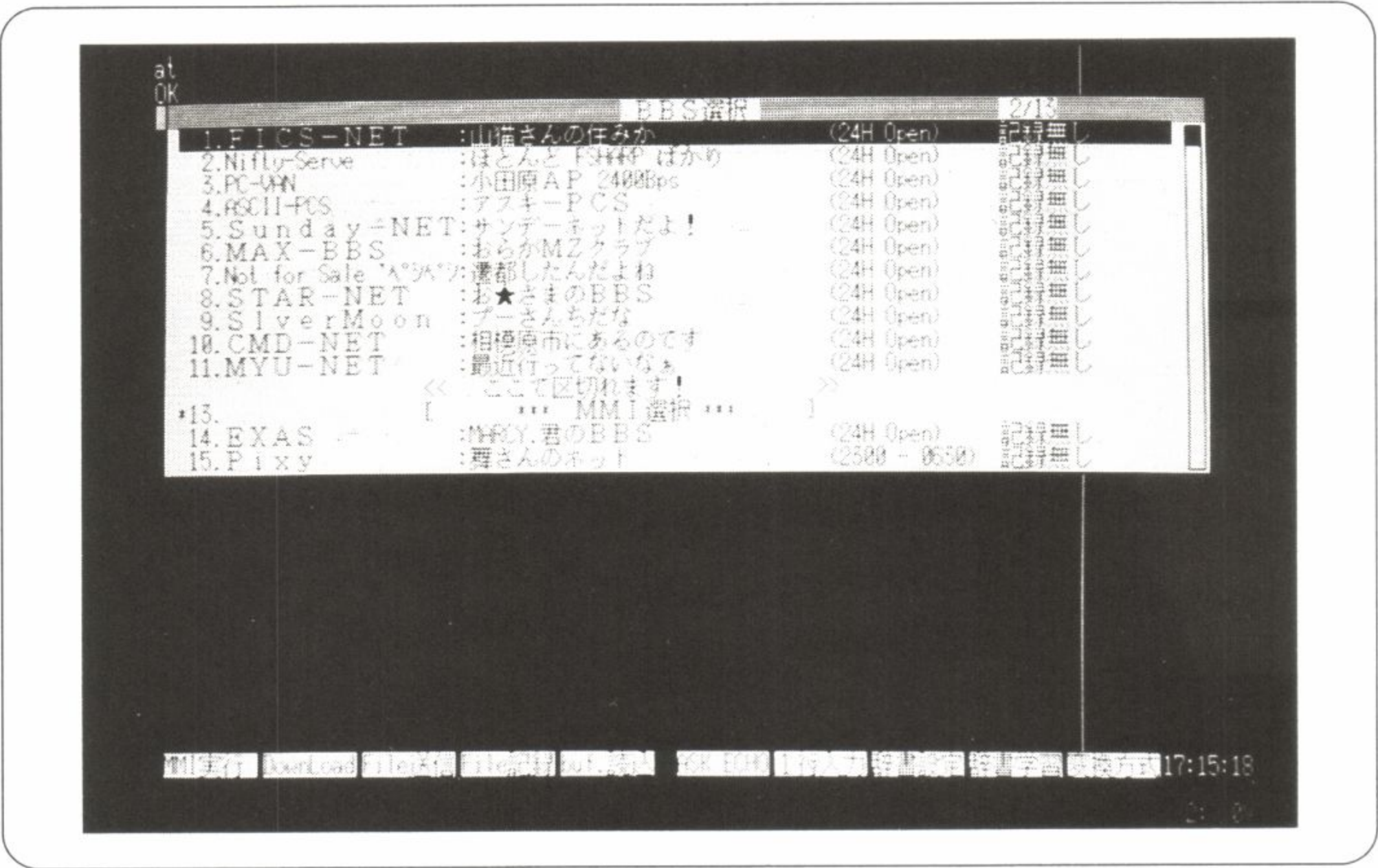
[↑][↓]のカーソルキーで項目を選んで[CR]キーを押すと、サブウィンドウが開きます。[設定] を選んで[CR]キーを押すと、設定するかどうかの確認ウィンドウが現れます。変更できる項目については、tmn.manを参照してください。

## 3. ホスト局へのアクセス

**●オートダイヤル／オートログイン**

TMNでは、MMIと呼ばれるマクロ言語を使ってオートダイヤル／オートログインを行います。MMIを記述したファイルがMMIファイルです（MMIの文法その他に関しては、付属のmmi.doc、tmn_mmi.man、tmn_mmiref.manを参照してください)。

実際にMMIを利用してホスト局にアクセスする場合は、まず[SHIFT]+[CTRL]+[P]キーか、[F1]キーを押してください。BBS-LIST(ファイル名はtmn.bbs)にサンプルとして登録されているホスト局の一覧が表示されます（Pht.4)。

Pht.4　BBS選択ウィンドウ

BBS-LISTは、ホスト局の名称や、そのホスト局にオートログインする際に使用するMMIファイルの名前が書かれたテキストファイルです。tmn.cnfで設定されているMMI-pathのディレクトリからMMIファイルを探して実行します。

[↑][↓]のカーソルキーを押すと、反転している部分（以下、この反転している部分も「カーソル」と呼びます）が動きますから、目的のホスト局にあわせて[CR]を押せば、モデムが自動的にダイヤルをかけはじめます。しかし、BBS-LISTに書かれているのは、あくまでもサンプルなので、このままでは使うことはできません。対応するMMIファイルを書き換える必要があります。

MMIファイルを書き換えるには、エディタを使って直接書き換えるか、MMIファイル自動作成用のウィンドウに必要事項を設定し、新たに作成する方法があります。ここでは自動作成を行ってみます。

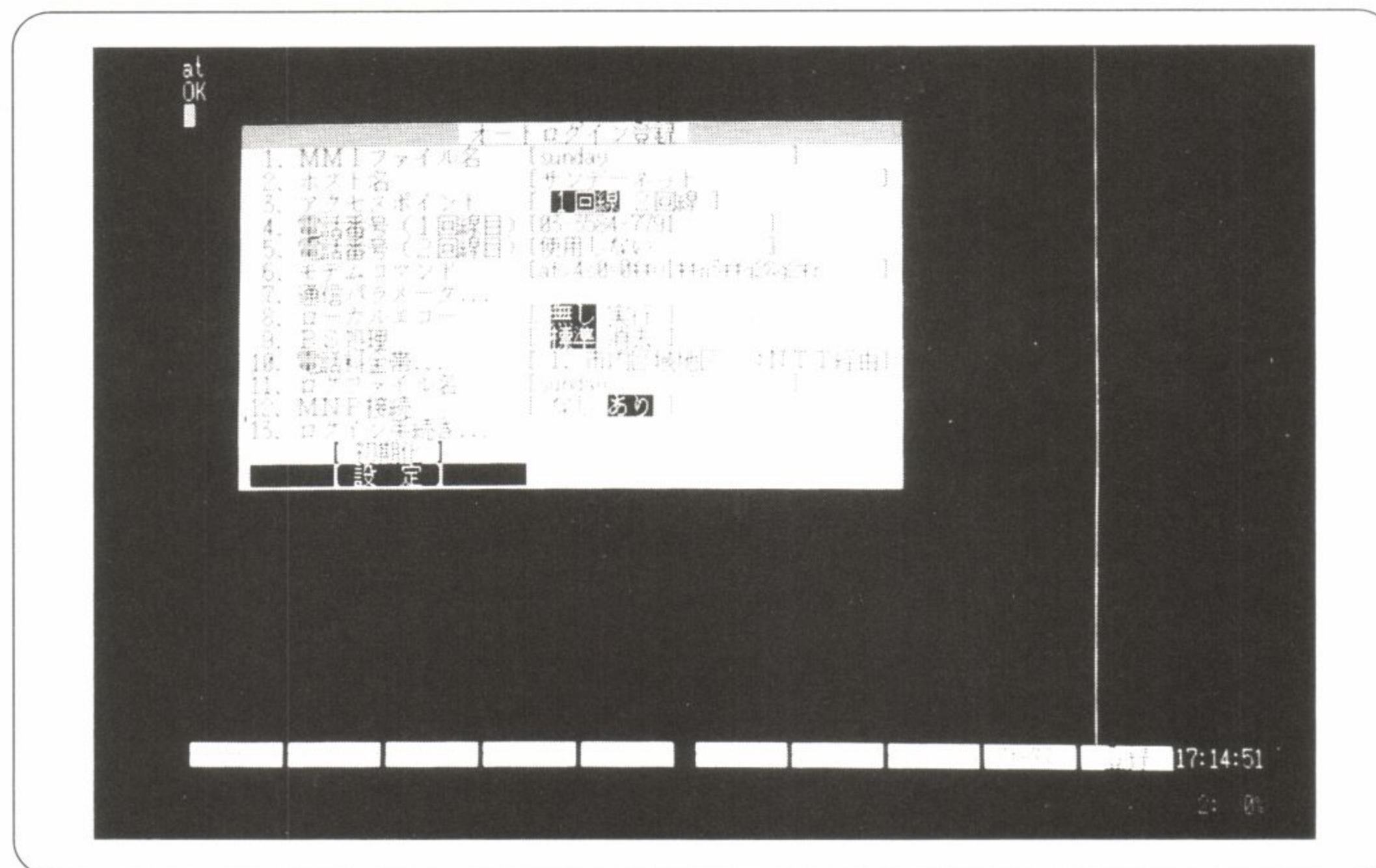
[ESC]キーを押して、[BBS選択]のウィンドウを閉じてください。次に、[OPT.2]+[F10]キーを押してください。MMIファイル作成用のウィンドウが開きます(Pht.5)。このウィンドウの中の項目を埋めていくと、基本的なMMIファイルが作成されます。

Pht.5　オートログイン登録ウィンドウ

　では実際に、サンプルにあるサンデーネット用のMMIを作ってみます。上下のカーソルキーでウィンドウ内のカーソルが動きますので、設定する項目を選んで[CR]キーを押してください。すると、入力用のウィンドウが出てきます。設定する項目は、以下のとおりです。

| 1．MMIファイル名 | MMIファイルの名称。拡張子はつけない<br>自動的に.mmiの拡張子がつけられる |
|---|---|
| 2．ホスト名 | ホストの名称 |
| 3．アクセスポイント | 2回線を選んだ場合、2つの電話番号に交互にダイヤルする |
| 4．電話番号（1回線目） | 1回線目の電話番号 |
| 5．電話番号（2回線目） | アクセスポイントで2回線を選んだときに有効になる<br>1回線目が話し中の場合、2回線目の電話番号と交互にダイヤルする |
| 6．モデムコマンド | 未設定の場合、tmn.cnfの設定になる |
| 7．通信パラメータ... | カーソルをここにあわせて[CR]キーを押すと、ウィンドウを開く<br>設定しなければ、TMNが現在動いている設定になる |
| 8．ローカルエコー | ホストによって変わるが、通常は［無し］ |
| 9．BS処理 | ホストによって変わるが、通常は［標準］ |
| 10．電話料金帯... | ホスト局までの距離を設定する<br>ここで設定した距離は、回線切断時に電話料金を計算するのに使われる。カーソルをあわせて[CR]キーを押すと、ウィンドウを開くので、ホスト局までの距離をカーソルで選んで[CR]キーを押して設定する(Pht.6) |
| 11．ログファイル名 | 通信中の記録をとるファイルを指定する<br>拡張子はつけないこと。自動的に.logの拡張子がつけられる |
| 12．MNP接続 | モデムによってどちらかを選択する<br>[なし]を選ぶと電話番号の末尾に“D”がつくが、このコマンドに対応していないモデムもある |
| 13．ログイン手続き... | ホストから送られてくる文字列と、それに対応して送信する文字列を設定する |

Pht.6　料金区域選択ウィンドウ

Pht.7　オートログインファイル作成ウィンドウ2

ひととおり設定したものがPht.7です。ログイン手続きは設定しません(まだどんな文字が送られてくるかわからないわけですから)。設定が終わったら、[設定]にカーソルをあわせて[CR]キーを押すと、確認のウィンドウが出てきますので、[Y]か[CR]キーを押すとMMIファイルが作成され、BBS-LISTに登録されます。MMIファイルが作成される場所は、tmn.cnfでMMI-pathが設定してあればそのディレクトリ、設定されていなければカ

レントディレクトリです。

ここで、あらためて[SHIFT]+[CTRL]+[P]か[F1]キーを押してください。ウィンドウの一番下に、今登録したホスト局の名前が出ているはずです。見つからなければカーソルキーでカーソルを下げてください。カーソルが画面の一番下まで行くとスクロールします。カーソルをあわせ、[CR]キーを押せばモデムが電話をかけはじめるはずです。運悪く回線がふさがっている場合は、十数秒待ってから再度自動的にダイヤルされます。もし、実行中のMMIを中断したいときは[ESC]キーを押してください。

また、複数のホスト局を選択しておいて回線がつながらない場合は、順に次のホスト局にダイヤルしていくこともできます。複数のホスト局を選択する場合は、[CR]のかわりに[SPACE]キーを押してください。ホスト局の名称の左に"!"の文字がつきます。再度[SPASE]キーを押すと、"!"が消えます。カーソルを移動させて複数のホスト局を選択したあと、[CR]キーを押せば上から順にダイヤルされ、回線がつながらなければ次のホスト局にダイヤルし、選択された最後のホストにもつながらなければ最初のホスト局に戻ってダイヤルしていきます。

ただし、この機能を使う場合、オートダイヤル用のMMIファイルにこの機能を使うためのコマンドが記述されていなければなりません。詳しくは、tmn_mmi.manを参照してください。自動作成用のウィンドウから作成したMMIファイルには、このコマンドが記述されています。

ホスト局に回線がつながるとMMIは終了しますから、あとはキーボードからID、パスワード等を打ち込んで、ホスト局にログインしてください。

ホスト局との接続を切ると、ウィンドウが出てきて通話時間と電話料金を表示します。また、MMIファイルの作成時にログファイル名を指定していますので、ホスト局に接続した時点から回線が切れるまでの間に受信した文字がファイルに記録されています。ファイルが書かれるディレクトリは、tmn.cnfのlog-pathで指定したディレクトリです(今は未指定なので、カレントディレクトリになります)。

### BBS-LISTの修正

BBS-LISTには、サンプルのホスト局がいくつか登録されていますが、そのホスト局にIDを持っていない場合は不要です。BBS-LISTはtmn.bbsというファイル名のテキストファイルですから、エディタで修正できます。

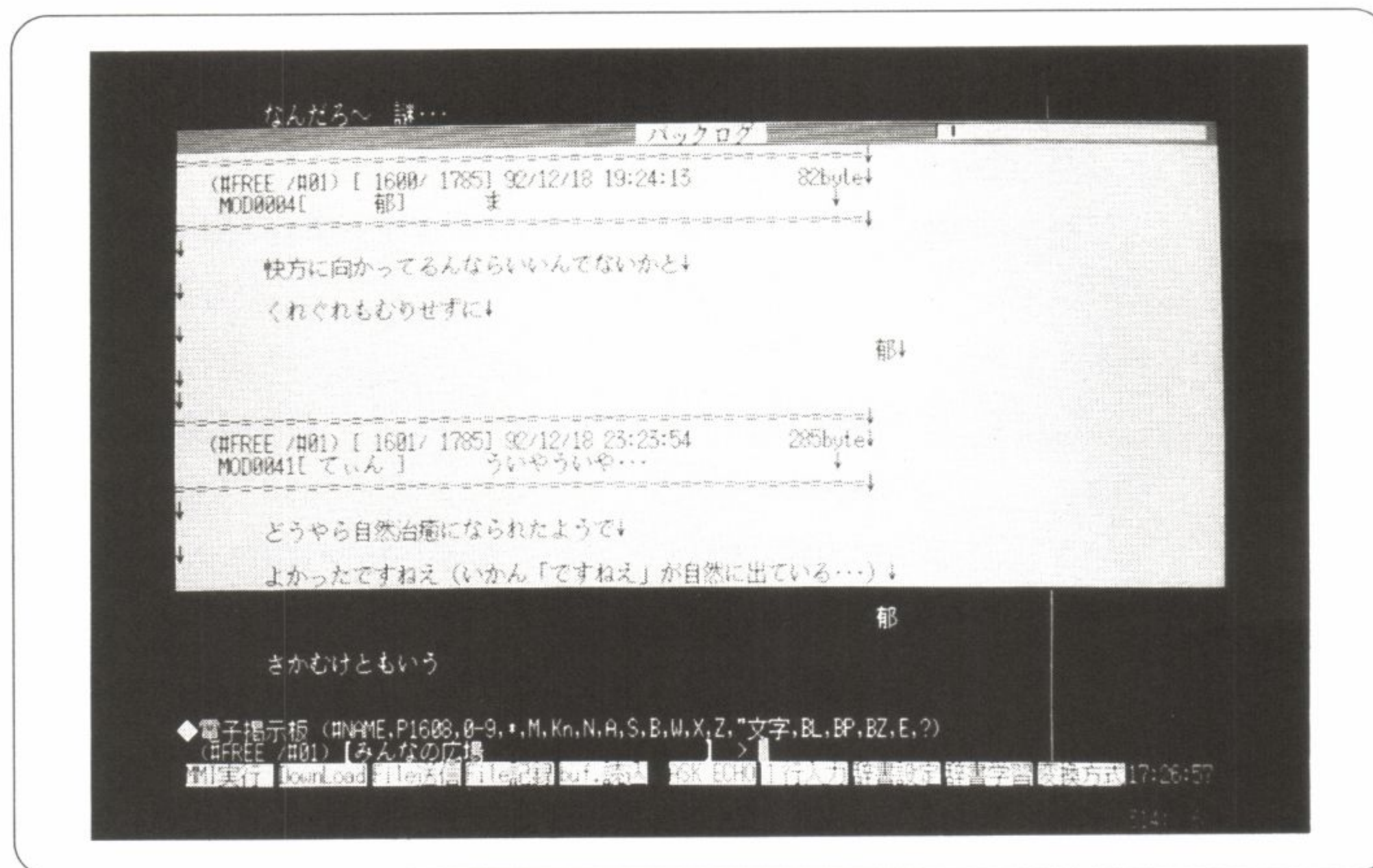
エディタでtmn.bbsを開くと、各項目の説明とともに、1行ごとにホスト局と対応するMMIファイル、ホスト局が開いている時間などがまとめられていますので、不要なものを削除したり、サンプルを参考に新しく追加したものを修正してください。

先ほど作成したMMIファイルはオートダイヤルまでで、ログインはキーボードから手で行わなければなりませんでしたが、MMIファイルをエディタで修正するか、自動作成用

のウィンドウのログイン手続きの項目を埋めて作成すれば、オートログインも可能です。

### ●バックログ

ホスト局に接続してから回線を切るまでの間、新しく文字をホストから受け取って改行していくたびに、以前に受信した文字は画面から消えていくわけですが、[ROLL DOWN]もしくは[UNDO]キーを押すと、バックログのウィンドウが開いて、これまでに受信した文字をさかのぼって見ることができます。

Pht.8　バックログウィンドウ

[ROLL　UP]、[ROLL　DOWN]キーで表示する領域が上下し、[OPT.1]、[OPT.2]、[SHIFT]キーのどれかを同時に押すことで表示のスピードが変えられます。

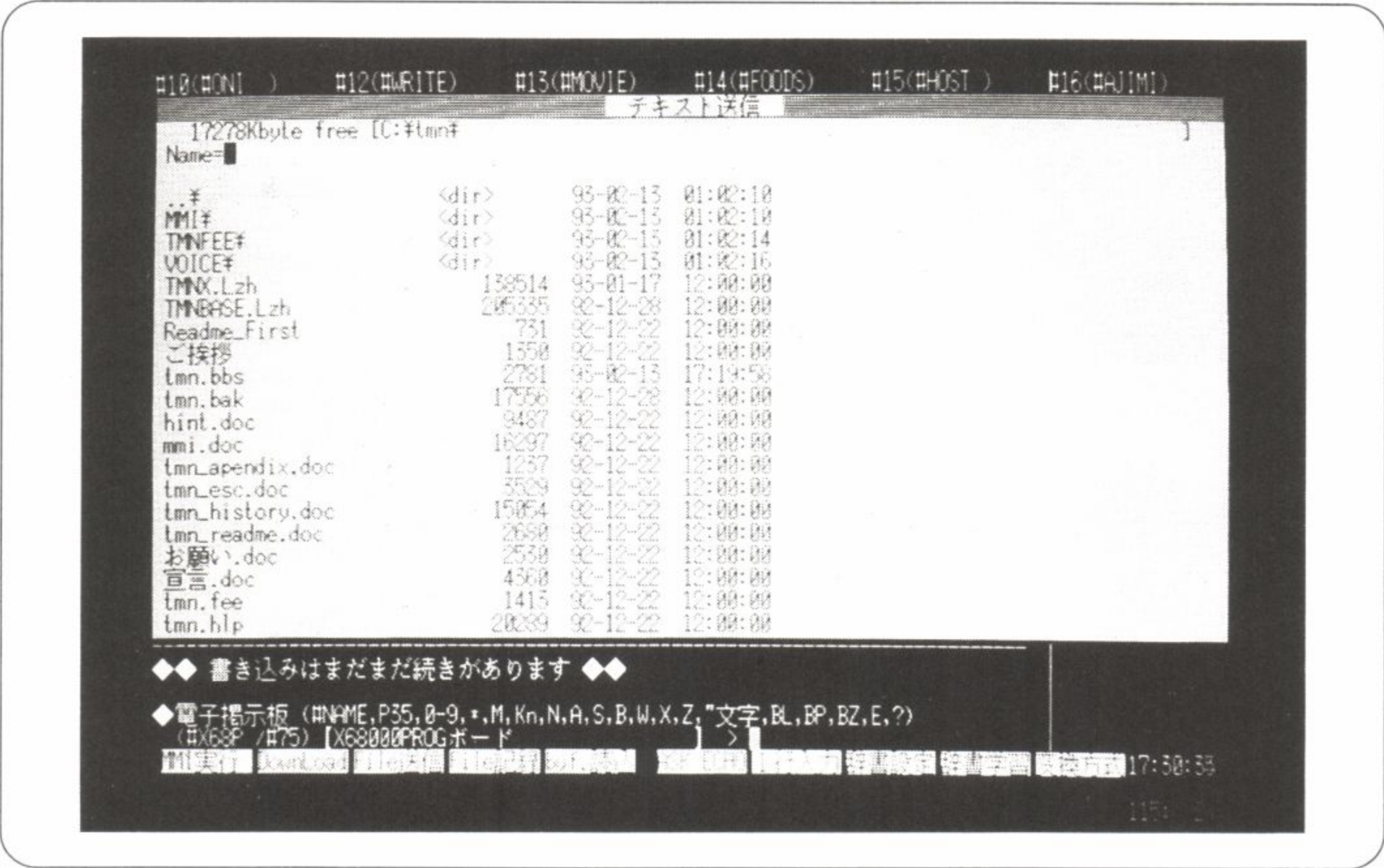
また、バックログの内容を修正してホスト局に送信したり、ファイルに落とすこともできます。これらの機能は回線が接続しているかいないかにかかわりなく使えますから、画面上から消えてしまった書き込みを読み直したり、そこから引用してホスト局に書き込むことができます。

### ●ファイル転送

#### ファイル選択

アップロードやダウンロードのときなど、ファイル名の入力が必要な場合は、[F3]キーを押して「テキスト送信ウィンドウ」を開きます(Pht.9)。直接キーボードからファイル名を入力するか、ウィンドウに表示されているファイル名をカーソルで指定することができ

ます。ドライブやディレクトリを変更する場合は特定のキー(Tbl.1参照)を押すと、選択用のウィンドウが開きます。

Pht.9 ファイル選択ウィンドウ

### 文字入力モード

「テキスト送信ウィンドウ」が開いた直後は、このモードです。ウィンドウ上部の"Name=" と書かれた部分にカーソルがあり、このままキーボードからファイル名を直接入力して指定することができます。

このモードでは、キーにTbl.1のような機能が割り当てられます。

Tbl.1 文字入力モードでのキーの機能

| キー | 機　　能 |
|---|---|
| [↑] | ドライブ選択モードへ移る |
| [↓] | ファイル選択モードへ移る |
| [←] | カーソル左移動／左端の場合はディレクトリ選択モードに移る |
| [→] | カーソル右移動 |
| [BS] | 1文字後退 |
| [DEL] | 1文字削除 |
| [ESC] | 処理をキャンセルする |
| [CR] | 決定 |

### ファイル選択モード

ウィンドウに表示されているファイルをカーソルキーで指定します。カーソルをあわせて[CR]キーを押せば、そのファイルが選択されます。Tbl.2以外のキーを入力すると、入力された順に文字を組み立て、それをファイル名と見なして順に検索していきます。

Tbl.2　ファイル選択モードでのキーの機能

| キー | 機　能 |
|---|---|
| [↑] | カーソル上移動／文字入力モードへ戻る |
| [↓] | カーソル下移動 |
| [←] | カーソル左移動（ロングのときは上移動） |
| [→] | カーソル右移動（ロングのときは下移動） |
| [ESC] | 文字入力モードへ戻る |
| [CR] | 決定 |
| [SPACE] | サーチのキャンセル |
| [HOME] | 表示フォーマットの切り替え |

### ドライブ／ディレクトリ選択モード

ドライブまたはディレクトリの一覧がサブウィンドウに表示されますから、カーソルをあわせて[CR]キーを押すと移動し、文字入力モードに戻ります。

Tbl.3　ドライブ／ディレクトリ選択モードでのキーの機能

| キー | 機　能 |
|---|---|
| [↑] | カーソル上移動 |
| [↓] | カーソル下移動 |
| [ROLL UP] | ページスクロールアップ |
| [ROLL DOWN] | ページスクロールダウン |
| [ESC] | 文字入力モードへ戻る |
| [CR] | 決定 |

### ●テキストファイルのアップロード

テキストファイルのアップロードは、[SHIFT]+[CTRL]+[T]、もしくは[F3]のキーに割り当てられています。キーを押すとファイル選択用のウィンドウが開きます。ウィンドウにはtmn.cnfのtype-pathで指定されたディレクトリにあるファイルが表示されます。type-pathが設定されていないときはカレントディレクトリにあるファイルが表示されます。

ファイルを選択すると（選択する方法は前述「●ファイル選択」を参照)、送信が始まります。送信中、一時的に止めたいときは[INS]キーを押してLEDを点灯させると停止します。再度押して、消せば再開、[ESC]キーで中断です。送信前から[INS]キーのLEDが点灯していると、停止したままになりますから注意してください。

アップロードしている間、ログファイルへの記録を停止したいときはtmn.cnfで“atlog-cut=on”としておけば、ログの記録は停止します。

ホスト局によっては送信の速度を調節しないとデータを取りこぼすことがあります。速度の調節はtmn.cnfで設定できますが、TMNを起動したあともコントロールパネルで変更することができます。

### ダウンロード

テキストファイルのダウンロードは、オートログイン用のMMIファイルに指定してお

けば、ホスト局に接続した時点から開始されます。それとは別のファイルに落としたいときは、[SHIFT]+[CTRL]+[O]、もしくは[SHIFT]+[F4]キーを押すとログファイルへの記録を中止します。[SHIFT]+[CTRL]+[L]、もしくは[F4]キーを押すと、ファイル選択ウィンドウが開きます。

ウィンドウには、tmn.cnfのlog-pathで指定されたディレクトリにあるファイルが表示されます。log-pathが設定されていないときは、カレントディレクトリにあるファイルが表示されます（選択する方法は「●ファイル選択」を参照)。また、すでに存在するファイルを選択すると、追加書き込みになります。

tmn.cnfで“off-log=close”と設定してあれば、回線が切れたときにログファイルを自動的にクローズします。

**●バイナリファイルのアップロード**

[SHIFT]+[CTRL]+[U]、もしくは[SHIFT]+[F2]キーでプロトコルを選択するウィンドウが開きます。カーソルキーかプロトコルの名称の横の番号を押すとカーソルが動きますので、[CR]キーを押すとファイル選択のウィンドウが開きます。tmn.cnfでup-pathが指定されていればそのディレクトリの、指定されていなければカレントディレクトリのファイルが表示されます。

### X-modem/X-modem-K/Y-modem[-g]/M-LINK

ファイルを選択すると送信状況を表示するウィンドウが開きます（選択する方法は「●ファイル選択」を参照)。

X-modemは、データチェックの方法にCRCかCHECK SUMを使うかで2種類に分かれ、またY-modem にも“-g”オプションによる高速転送がありますが、どのプロトコルが使われているかはTMNが判定するので、指定の必要はありません。

### Y-modem[-g] BATCH

バッチモードを選択したときは、ワイルドカードでファイルを選択します。ファイル名の最後に“¥”をつけずにディレクトリを指定した場合は、そのディレクトリ内のファイルをサブディレクトリを含めてすべて転送します。また、“-g”オプションはTMNが判定します。

### B-plus

B-plusは大手商用ネットのNIFTY-Serveで使われているプロトコルです。B-plusではホスト局からファイル名の入力を求められますので、アップロードしたいファイル名をキー入力します。あとはホスト局から送られてくる開始コードを認識してTMNが転送を

自動的に始めます。

●バイナリファイルのダウンロード

[SHIFT]＋[CTRL]＋[D]、もしくは[F2]キーでプロトコルを選択するウィンドウが開きます。アップロードのときと同様にプロトコルを選択してください。このとき開くファイル選択ウィンドウはディレクトリ／ファイルを指定する1行だけのものですが、キーの機能は通常のファイル選択のウィンドウと変わりません。tmn.cnfでdown-pathが指定されていた場合はそのディレクトリ、指定がない場合はカレントのディレクトリが表示されます。

X-modem/X-modem-K

X-modemでは、ダウンロードのときもファイル名を指定する必要があります。ファイル名を指定すると、受信状況を表示するウィンドウが開きます（選択する方法は「●ファイル選択」を参照）。

X-modemではデータのチェック方法にCRCかCHECK SUMの2種類が使われ、TMNが自動的に判定することが可能です。ただし、相手側のホスト局がCHECK SUMしか使えない場合、判定に要する時間（わずか数秒ですが）が無駄になります。あらかじめCHECK SUMしか使えないとわかっている場合は、tmn.cnfで“xmodem-type=sum”と設定しておくと、CHECK SUMのみを使用し、判定の時間が省けます。

Y-modem[-g]/M-LINK

受信したファイルを置くディレクトリを指定すると、受信状況を表示するウィンドウが開きます。ディレクトリを指定するときは最後に必ず“¥”をつけてください。

B-plus

アップロード同様、ファイル名を入力すると、ホスト局と受信開始のやりとりを行い、受信を自動的に開始します

●Flying X-MODEM

MNPやLAPMなどの機能を持ったモデムを使ってX-modemでファイルの転送を行うと、回線上の速度に比べ、実際の転送速度はかなり遅くなります。X-modemでは、データを128バイトに区切って送り、受信側からの正常に受信できたという返事を待って次のデータを送っているのですが、MNP、LAPMでもモデム間でほぼ同様のことを行っています。

モデムは、ある程度のデータをパソコンから受け取るか、データの流れが一定時間とぎ

れるとデータを送信します。X-modemが送るブロック（送信するデータと、正常に受信できたかどうかを検査するためのコードなどを含んだ、一区切りのデータ）のサイズと、モデムがまとめて転送するサイズとがあわないので、モデムが相手側のモデムにデータを送り出すまでに余分な時間がかかり、全体の速度が落ちてしまいます。

また、受信の確認に使うコードは1バイトですから、ここでも時間が余分にかかります。Y-modemでも同様ですが、Y-modemの場合はブロックのサイズが大きいため、X-modemほどは効率が悪くありません。

これを避けるために、送信時は正常に受信されるものと仮定して、あらかじめいくつかのブロックを先送りしておく、受信時は正常に受信できると仮定して、正常に受信できたときのコードを先送りしておくのが、Flying X-MODEMです。

TMNでは、X-modem使用時にFlyingを行うか、また、行う場合はどれくらいの数を先送りするかをコントロールパネルから設定できます。コントロールパネルの環境数値設定ウィンドウ内のFlying X-MODEM 指数が0ならば通常のX-MODEM、0でなければその数だけブロックを先送りします。

ただし、これはあまり一般的な方法ではないので、ホスト局によってはうまく動かないことがあります。

## 4．カスタマイズ

TMNをより使いやすくするためには、あなたの好みや環境にあわせてカスタマイズしてください。それにはコンフィグファイルの内容を変更することが必要です。コンフィグファイルは、TMNが起動するときに参照する各種の設定が書かれているファイルで、ファイル名はtmn.cnfです。

tmn.cnfはテキストファイルですので、ed.xなどのエディタを使って変更できます。tmn.cnfにはデフォルトの設定とともに簡単な説明が書かれていますが、設定に関する詳しい説明はtmn_conf.manを参照してください。

また、つねにTMNを展開したディレクトリをカレントディレクトリにしなくてもすむように展開したディレクトリにパスを通し、tmsio.xもAUTOEXEC.BATやCONFIG.SYSなどから常駐させるようにしたほうがよいでしょう。

具体的には、次の例のようにAUTOEXEC.BATに追加します。

```
PATH A:¥;A:¥USR¥BIN;....;A:¥TMN ←TMNを展開したディレクトリをパスに追加
     └────────┬────────┘
           元のパス
tmsio                              ←tmsioの常駐
SET tmncnf=A:¥TMN¥tmn.cnf
```

tmsioはtmn.xと同じディレクトリにある必要はありませんので、パスの通っている他のディレクトリに移してもかまいません。tmsio.xのオプションの詳細はtmsio029.lzhに含まれているドキュメントを参照してください。

環境変数tmncnfはTMNの設定ファイルの指定です。

なお、A:¥TMN以外のディレクトリにファイルを展開した場合は、適宜変更するようにしてください。

**●tmn.cnfの変更**

TMNは起動時および実行中に各種のファイルを参照します。それらのファイル名や、どのディレクトリに収められているかをコンフィグファイルtmn.cnfで指定します。ここでは、特に断りのない限り、A:¥TMNにファイルが展開されたものとして説明していきます。

MMIファイルのパス　（添付のtmn.cnfではMMI-path=.¥mmi¥＊.mmi）
MMI-path=A:¥TMN¥mmi¥＊.mmi

TMNでは、MMIと呼ばれるマクロ言語を使うことによってオートパイロットが可能です。このMMIを記述したファイルがMMIファイルです。ウィンドウから一覧を表示して実行するか、TMN内部のコマンドラインからファイル名を直接指定することができます。

そのとき、ファイルを検索するディレクトリをここで指定します。文字列の最後は“¥”か、ワイルドカードで指定してください。

BBS-LISTの指定　（添付のtmn.cnfではnetsel-file=.¥tmn.bbs）
netsel-file=A:¥TMN¥tmn.bbs

MMIを使用するとオートダイヤル、オートログインが可能です。そのMMIファイルをBBS-LISTと呼ばれるファイルに登録することで、メニューウィンドウから選択し、実行することができます。netsel-fileは、そのファイルの指定です。

ヘルプファイルのパス　（添付のtmn.cnfではhelp-file=.¥tmn.bbs）
help-file=A:¥TMN¥

実行中に表示可能なヘルプファイル（TMNBASE.LZHに含まれるtmn.hlp）のあるディレクトリを指定します。

電話料金リストの指定　　　　　　　（添付のtmn.cnfでは、fee-list=.¥tmn.fee）
fee-list=A:¥TMN¥tmn.fee

　TMNには電話回線の接続時、および切断時に電話料金を計算する機能があります。計算時に使用する地区別の電話料金の一覧が書かれたファイルを指定します。

電話料金ログの指定　　　　　　　（添付のtmn.cnfではfee-log=.¥telfee.log）
fee-log=A:¥TMN¥telfee.log

　回線切断時に、接続していたホスト局、時間、電話料金などを記録するファイルを指定します。

アップロードファイルのパス　　　　（添付のtmn.cnfではup-path=¥up¥）
up-path=

　X-modemなどのプロトコルを使って送信するときに、転送するファイルを検索するディレクトリです。無指定のときはカレントディレクトリになります。

ダウンロードファイルのパス　　　　（添付のtmn.cnfではdown-path=¥down¥）
down-path=

　XMODEMなどのプロトコルを使って受信するときに、転送されたファイルを書き込むディレクトリです。無指定のときはカレントディレクトリになります。

オートタイプファイルのパス　　　　（添付のtmn.cnfではtype-path=¥type¥）
type-path=

　テキストファイルを送信するときに、転送するファイルを検索するディレクトリです。無指定のときはカレントディレクトリになります。

ログファイルのパス　　　　　　　（添付のtmn.cnfではlog-path=¥log¥）
log-path=

　通信中に受信した文字を記録しておくファイルのあるディレクトリです。無指定のときはカレントディレクトリになります。

MMI-path、up-path、down-path、type-path、log-pathはTMNの起動後にコントロールパネルからも変更できます。

また、tmn.cnfの中には、この他にも各種のファイルネームを設定するところがありますが、これらについての詳細はtmn_conf.manを参照してください。ファイルを指定するときはフルパスで指定してください。

### ●モデムおよびRS-232Cの設定

起動時のRS-232Cの設定

rs-mode=9600n81rns　　　　(添付のtmn.cnfではrs-mode=9600n81rn)

rs-modeはRS-232Cの設定です。これが設定されていない場合、Human68kの設定そのままになります。使用するモデムや、接続するホスト局によって異なります。変更する場合は、以下の項目を見て変更してください。

| | |
|---|---|
| 9600 | 通信速度(75、150、300、600、1200、4800、9600、19200、38400) |
| n | パリティビット(n:なし　e:偶数　o:奇数) |
| 8 | ビット長(5、6、7、8) |
| 1 | ストップビット(1、2) |
| r | フロー制御(n:なし　x:Xフロー　r:RS/CS制御　w:X＋R両制御) |
| n | シフトイン／アウト(n:なし　s:あり) |
| a | シフトJIS固定モード(a:自動判別　s:固定) |

パソコン通信で使用する場合は、通信速度とフロー制御を除いて特に変更しなくてもかまいません。通信速度、フロー制御はモデムによって変わりますので、モデムのマニュアルも参照してください。

また、TMNで使用するRS-232Cドライバのtmsio.xは、受信時に新JIS、旧JIS、シフトJIS、NEC-JISの自動判別ができます。“a”を指定すれば自動的にシフトJISに変換してTMNにデータを渡しますが、その分、遅くなります。高速なモデムを使い、TMNの動作が間にあわないときは“s”を指定してください。

モデムのダイヤルコマンド

modem-dial=atdp　　　　(添付のtmn.cnfと同じ)

modem-dialは、マクロによってオートダイヤルを行うときにモデムに送るコマンドを指定します。ATモデム（第1章を参照）ならばATDP(ダイヤル回線)か、ATDT(トーン回線）のどちらかを設定してください。

モデム初期化コマンド

modem-init=atx4s0=0¥¥v1¥¥n3¥¥q2&q2¥r　（添付のtmn.cnfと同じ）

　modem-initは、オートダイヤル時にモデムに自動的に送られるコマンドを指定します。これによってモデムの動作が決まります。コマンドに“¥”が含まれるときは、“¥¥”と2つ続けてください。最後の“¥r”はCR(キャリッジリターン)を意味します。

　あらかじめ、tmn.cnfに設定されているコマンドは、

| | |
|---|---|
| x4 | リザルトコードを拡張モード4にする |
| s0=0 | 自動着信を禁止する |
| ¥vl | リザルトコードを単語形式にする |
| ¥n3 | オートリライアブルモードに設定 |
| ¥q2 | MNPモード時にシリアルポートフロー制御をrs/csで行う |
| &q2 | 非MNPモード時にシリアルポートフロー制御をrs/csで行う |

という意味になりますが、この設定はモデムによって変わります。モデムのコマンドについては、モデムのマニュアル、ないしは第1章を参照してください。

**●スクロールバッファの設定**

max-lines=g　（添付のtmn.cnfと同じ）

　スクロールバッファが設定されていると、いったん画面に表示されたのち、画面から消えてしまった文字をウィンドウ上でさかのぼって表示することができます。max-linesは、そのためのバッファの大きさを指定します。max-lines=1000のようにバッファの大きさを行数で指定するとメインメモリ上にバッファが確保され、“g”を指定するとグラフィックVRAM上に確保されます。RAMディスクなどでグラフィックVRAMを使用していると、スクロールバッファをグラフィックVRAMに設定することはできません。行数で指定した場合、メインメモリの空きが許す限りの大きさが指定できます。

　なお、スクロールバッファは最初と最後がつながっているリング形式になっていますので、最後までいくと最初の行に戻って上書きされていきます。

**●その他の設定**

時間経過メッセージの設定

time-mode=on　（添付のtmn.cnfと同じ）

time-wait=10　（添付のtmn.cnfと同じ）

voice-mode=on　（添付のtmn.cnfと同じ）

TMNは、起動／終了時や、実行中一定時間RS-232Cから入力がなかったり、キーが押されないとPCM機能を使用して独り言を言いだします。気になる場合は“time-mode=off”と変更してください。独り言の間隔は“time-wait”で変更できます。

また“voice-mode=off”とすれば、音声でなく、文字でメッセージを表示します。この場合は、¥TMN¥voiceディレクトリにある音声データは不要になるので削除してもかまいません。

画面の幅の指定

width=96　　　　　　　（添付のtmn.cnfと同じ）

桁位置ゲージの指定

scale-pos=80　　　　　　（添付のtmn.cnfと同じ）

scale-line=on　　　　　　（添付のtmn.cnfと同じ）

TMNの画面の幅は最大で半角96桁です。しかし、パソコン通信に使われているパソコンの画面は80桁表示が一般的ですから、画面幅いっぱいの文章を送ると、他機種では読みにくくなります。

そこで、画面上にガイドラインを出して、書き込み時の目安とすることができます。これはあくまでも目安ですから、ガイドラインを越えても改行されることはありません。scale-posで桁数を指定し、scale-lineで表示のon/offを指定できます。

以上で基本的な設定は終了です。しかし、ここまでで設定した項目は、tmn.cnfのごく一部でしかありません。この他にも使用する環境や好みにあわせて設定を変えられますので、tmn_conf.manをじっくりと読んでみてください。

**【その他】**…………………　同梱されているhint.docにTMNを使ううえでのヒントが書かれていますが、若干補足しておきます。

**●メインメモリの空きが少ない場合**

TMNBASE.LZHに含まれているtmn.cnfをそのまま使うと、およそ600Kバイトから750Kバイト程度のメモリを必要とします。実行中にメモリが足りなくなると機能が制限される場合があります。そのようなときは、以下のように設定を変えてみてください。

**独り言**

TMNはPCMを使って、起動／終了時や一定時間入力がないときなどに喋りはじめま

す。そのためのデータをメモリ上に持つため、80Kバイト弱必要とします(TMNBASE.LZHに付属のデータを使用する場合)。独り言を言わないように設定すれば、その分のメモリが不要になります。

```
voice-mode=off
```

もちろん、PCMデータを収めたVOICEディレクトリの内容は不要となりますから、ディスクスペースに余裕のない場合は消してしまってもかまいません。

電話料金加算時と、毎時00分にtmn.cnfで指定したPCMファイルの内容を鳴らすことができますが、独り言同様、その分のメモリを消費します。あまり大きいPCMファイルは指定しないほうがよいでしょう。

### MMIスタックの設定

MMIはスタックの操作を基本としたFORTHに似た処理体系を持つ簡易言語です(詳しくは、付属のmmi.docを参照)。通常TMN内部に128段のスタックが用意されますが、オートログインに使う程度では16段ほどで十分です。

```
stack-len=16
```

と16段に設定すれば、その分メモリを消費しません。ただし、複雑なMMIを動かす場合はスタックが不足するため、実行できなくなることがあるかもしれません。

voice-modeの設定と上記の設定で100Kバイト程度メモリに余裕ができます。

### バックログバッファの指定

tmn.cnfのmax-lines=でバックログの大きさを行数で指定すると、その分メインメモリを消費します。"g"を指定するとバッファはグラフィックVRAM上に置かれますので、使用するメインメモリの量は減りますが、グラフィックVRAMをRAMディスクとして使えなくなります。メモリが少ない場合、どちらを犠牲にすべきか悩むところです。かといって、バックログを使用するのとしないのとでは使用感がまるで違います。どうしようもないときはあきらめてメモリを増設しましょう。

●TMN終了後やチャイルドプロセスを実行したときに[SHIFT]、[CTRL]がきかない

FEPやキー入力をフックする常駐プログラムとの相性でまれに起こりますが、このときは、[SHIFT]や[CTRL]キーを数回押すと正常に戻ります。

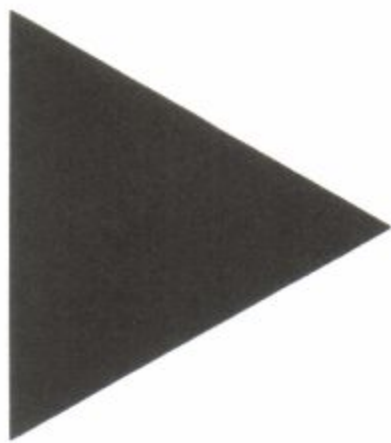
●オプションスイッチに“-g”を指定していても画面のバックにCGが表示されない

バックグラウンドの表示色が0以外ではグラフィックが表示されません（X68000のハードウェア上の制約です)。tmn.cnfで“back-color=00,00,00,0”と指定してください。

●tmn.cnfで“atlog-cut=on”としてもアップロードしたファイルの一部がログに残る

MNP、LAPMの機能を持ったモデムを使用した場合、ホストに送信した文字がエコーバックされるまでに若干の時間差があります。アップロードが終了したあとも、その内容がホストから送られてくるからです。

●ホスト局との回線が切れても電話料金を計算しない

RS-232CのCD信号を、キャリア信号を検出時のみONになるように設定してください。接続時間を計るためにモデムのCD信号をチェックしているためです。

# MuTerm.X

## コンパクトで高速・高機能なターミナルソフト

【概要】…………………… 比較的容易な設定で使える高機能ターミナルプログラムです。その特徴を挙げてみましょう。

(1) PC-9801の通常のターミナルプログラムの場合、バックログ（後述）は32Kバイト程度だが、MuTermを使うと、X68000のグラフィックVRAMを使用して512Kバイトまでバックログをとることができ、メインメモリの場合だと、フリーエリアがあるだけバックログをとることができる。
(2) 最高通信速度は38400bpsまで可能。
(3) X68000標準の画面モードには存在しない、640×400ドットの画面モードを使用して、PC-9801と互換性のある通信画面を実現している（エスケープシーケンスもPC-9801互換）。
(4) オートダイヤル、リダイヤル、オートログイン機能を備えている。リダイヤルしながら（メモリがあれば）チャイルドプロセスを呼び出し、文章の編集、ゲーム等もできる。
(5) XMODEM、YMODEM、M-LINKプロトコルで送受信できる。
(6) 半角カタカナを半角ひらがなに変換して表示することができる。
(7) 漢字フォント用ファイルを用意することにより、いわゆる98文字(①、Ⅱなど)を表示できる。

他にも、MuTermは大変優れた機能を備えており、非常にコンパクトで、起動、終了が敏速です。

【作者】…………………… はちくん　サンデーネット sun1387

【作者からの言葉】…… この通信ソフトを作ろうと思ったのは、３年半ほど前、当時通信初心者だった私が、さまざまな色や属性のついた文字を送ることのできるエスケープシーケンスの魅力にとりつかれたことから始まります。これを自分のX68000で見てみたいと考えていたとき、「X68はハード的に文字の点滅をサポートしていないので、完全なエスケープシーケンスを実現するのは困難」ということを聞き、それならソフトウェアでどうにかならないか？　と思ったわけです。

そこで、さっそく通信ソフトを作ってみたのですが、そのときは、ただ受信した文字を画面に表示するだけの機能しか持っていませんでした。しかし、それでもなんとか仕様に近い表示ができるようになったので、他の68ユーザの方にもこの環境を味わって欲しいと考え、電子メイルを使って少数の親しい方々に配布しました。

そのあと少しずつ機能を追加していき、それにともなって配布する方々もだんだん増えてくると、フリーソフトにしてみたらどうだろう、というご意見をいただくようになりました。18回目のバージョンアップを機についに一般公開に踏み切り、現在のバージョンに至ったというわけです。

ちなみに、最近は大学のほうが忙しく、個人的なプログラミングの時間があまりとれず、バージョンアップ作業はやむなく休止しておりますが、そのうちにまとまった時間がとれれば、もう少し高度な使い方にも耐えるものを目指して頑張ってみようかと思っています。

**【推薦します】**…………　なにしろ軽くて速いのが売りだと思います。なんとメモリを400Kバイト弱しか食わない。それでも通信するのに必要な機能はひととおり揃っています。初心者やメモリが少ない人、特にフロッピーベースで使っている人は、とりあえずこれで通信を始めてはいかが？

MAX BBS　てぃん

**【インストール】**………　まず、MUTERM.LZHをLHA.Xなどで次のように解凍します。

LHA X MUTERM.LZH[CR]（.LZHという拡張子は省略可能）

すると、次のようなファイル群ができあがります。

```
MUTERM    X     51200  92-03-03  12:00:00
MUTERM    CNF    2597  92-03-03  12:00:00
MUTERM    DOC    7526  92-03-03  12:00:00
MUTERM    MAN   60022  92-03-03  12:00:00
GETFONT   COM    4028  92-03-03  12:00:00
GETFONT   DOC    4065  92-03-03  12:00:00
ESC       DOC    2880  92-03-03  12:00:00
```

このファイル群を適当なディレクトリやディスクにコピーし、パスを通すだけで使えるようになります。

**【使用する前に】**………　なお、MUTERM.FNT があると、俗に言われる98文字をMuTermの上で見ることが

できるようになります。手元にPC-9801がある方(あるいは、まわりにPC-9801を持っている人がいる方)は利用してみてください。MUTERM.FNTの作成方法はドキュメントを見てください。

**【使用法】**

最初に、モデムをX68000と正しく接続します。電話線はモデムのLINE端子に、X68000とモデムをRS-232Cケーブルでつなぎます[注]。

注：モデムの設定や接続方法は第1章をご覧ください。

次に、モデムの電源を入れて、MuTermを立ち上げます。

MUTERM[CR]

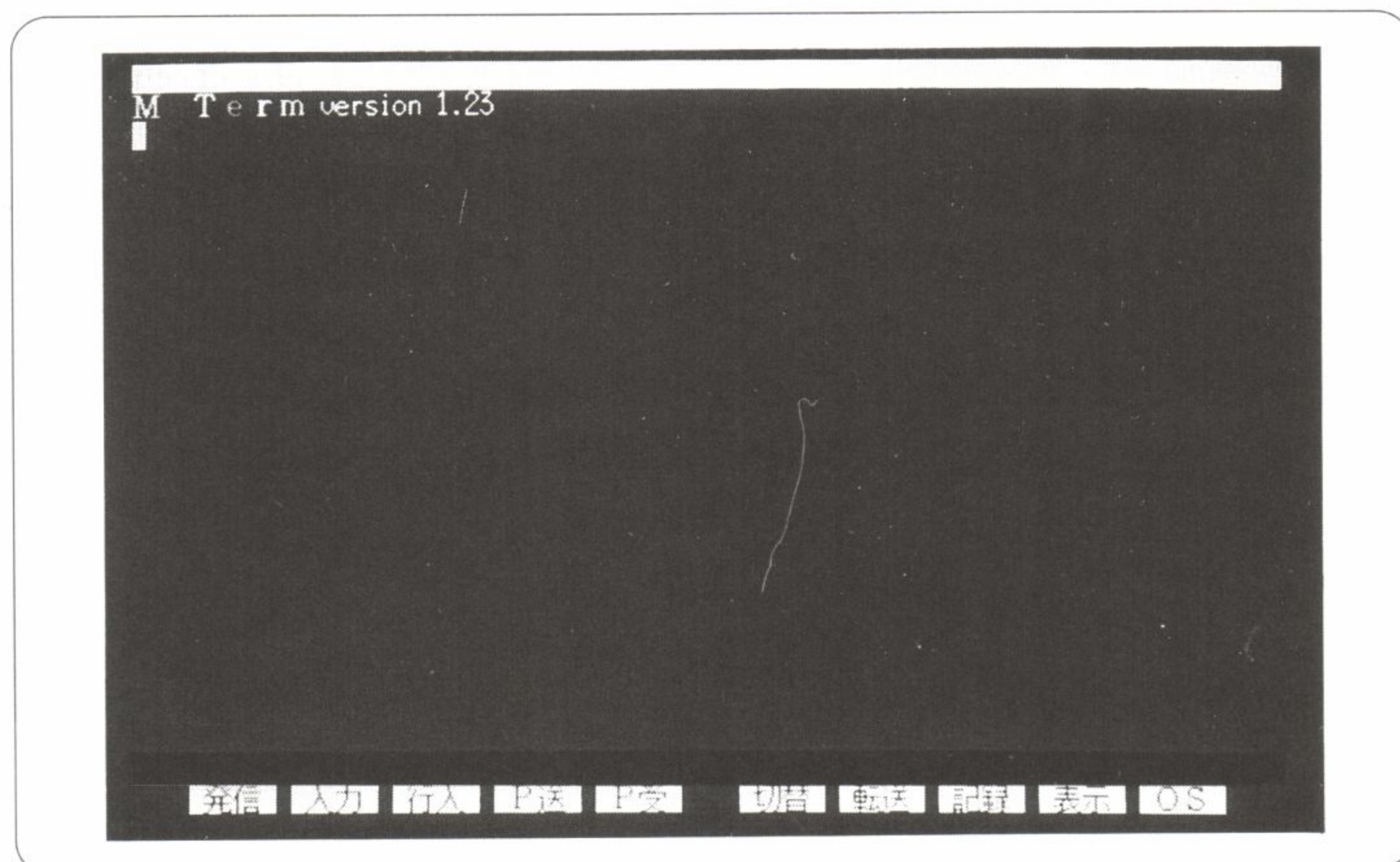
MuTermの起動に先立ち、特にRS-232Cドライバ等の組み込みや、設定ファイル名等を指定する必要はありません。MUTERM.Xと同じディレクトリにあるMUTERM.CNFが設定ファイルとして読み込まれます。

もちろん、別の設定ファイル名を指定することも可能です。この場合は、

MUTERM 設定ファイル名[CR]

とします。

起動すると、次のような画面になります(Pht.1)。

Pht.1 MuTermの起動画面

ここで、モデムが正常につながっているかどうかを確認するために、

AT[CR]

と打ち込みます。そして、

OK

と返ってきたら、モデムとX68000は正常につながっていることになります。

**【オプション】**…………　MuTermは、起動時に以下のようなオプションスイッチを指定することができます。

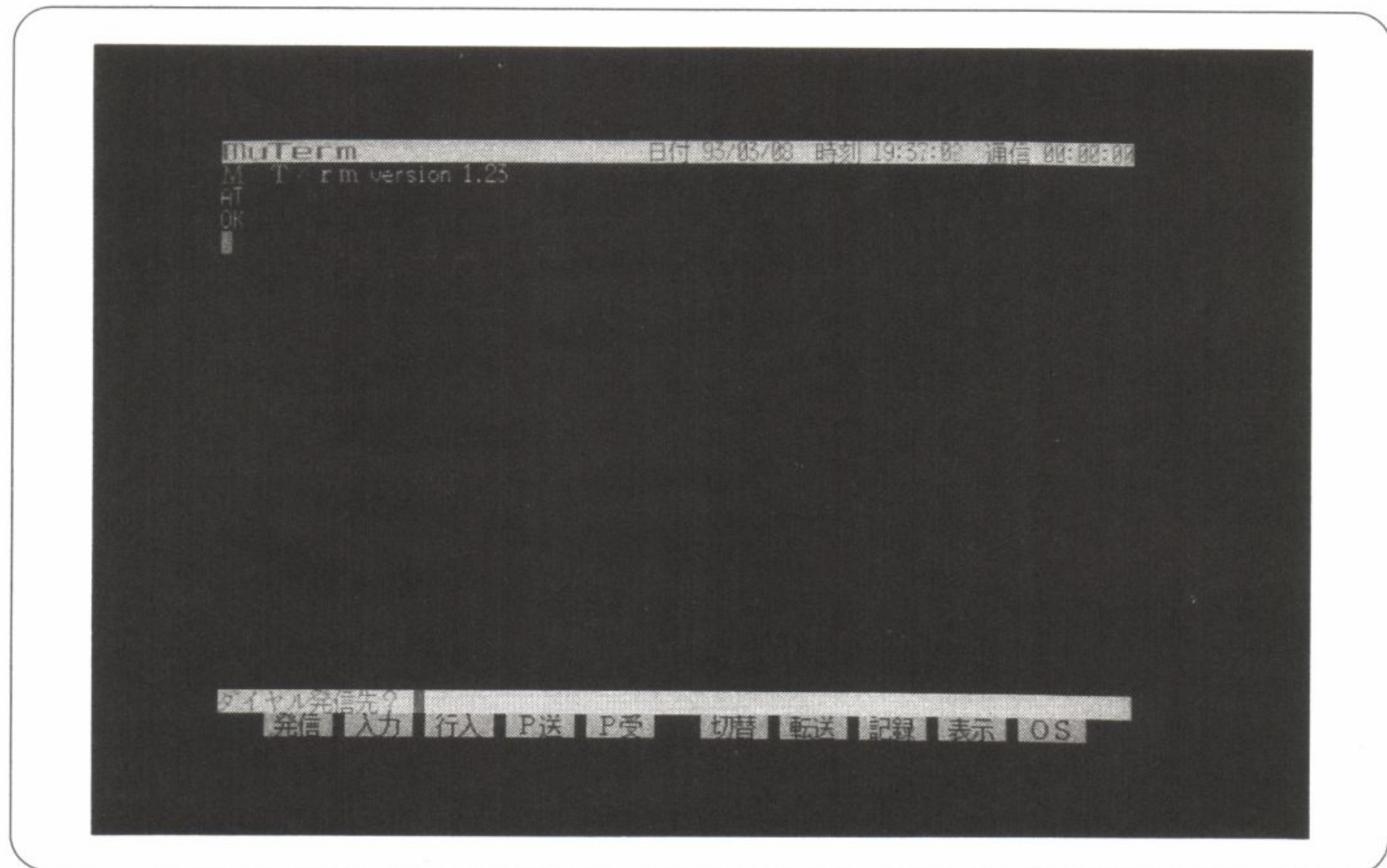
| オプションスイッチ | 内　容 |
|---|---|
| /a | オートダイヤル発信（1〜）<br>CNFファイル内のADIALに相当 |
| /b | バックログバッファ（4〜）<br>CNFファイル内のBLBUFに相当 |
| /c | 時計表示の有無（0〜59）<br>CNFファイル内のCLOCKに相当 |
| /h | ヒストリライン数（1〜）<br>CNFファイル内のHISに相当 |
| /m | FM音源使用（0/1）<br>CNFファイル内のMUSICに相当 |
| /r | 受信バッファ（1〜255）<br>CNFファイル内のRSBUFに相当 |

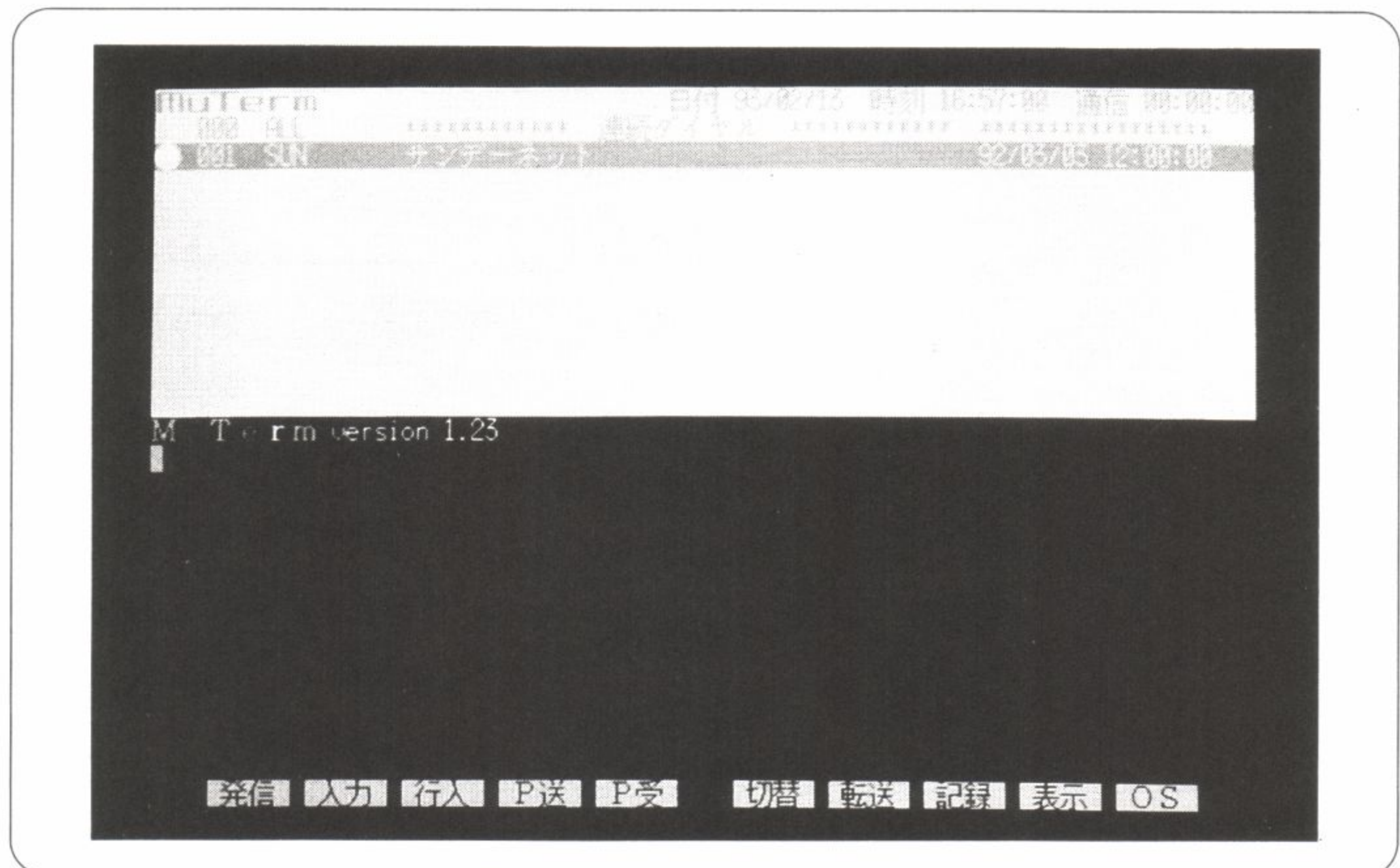
注：オプションスイッチに関する詳細はMUTERM.MANを参照のこと。

**【主な使用法】**…………　MuTermの機能説明を、MUTERM.CNFにすでに登録されているサンデーネットに実際にログインする場合を紹介しながら、順に説明していきます。

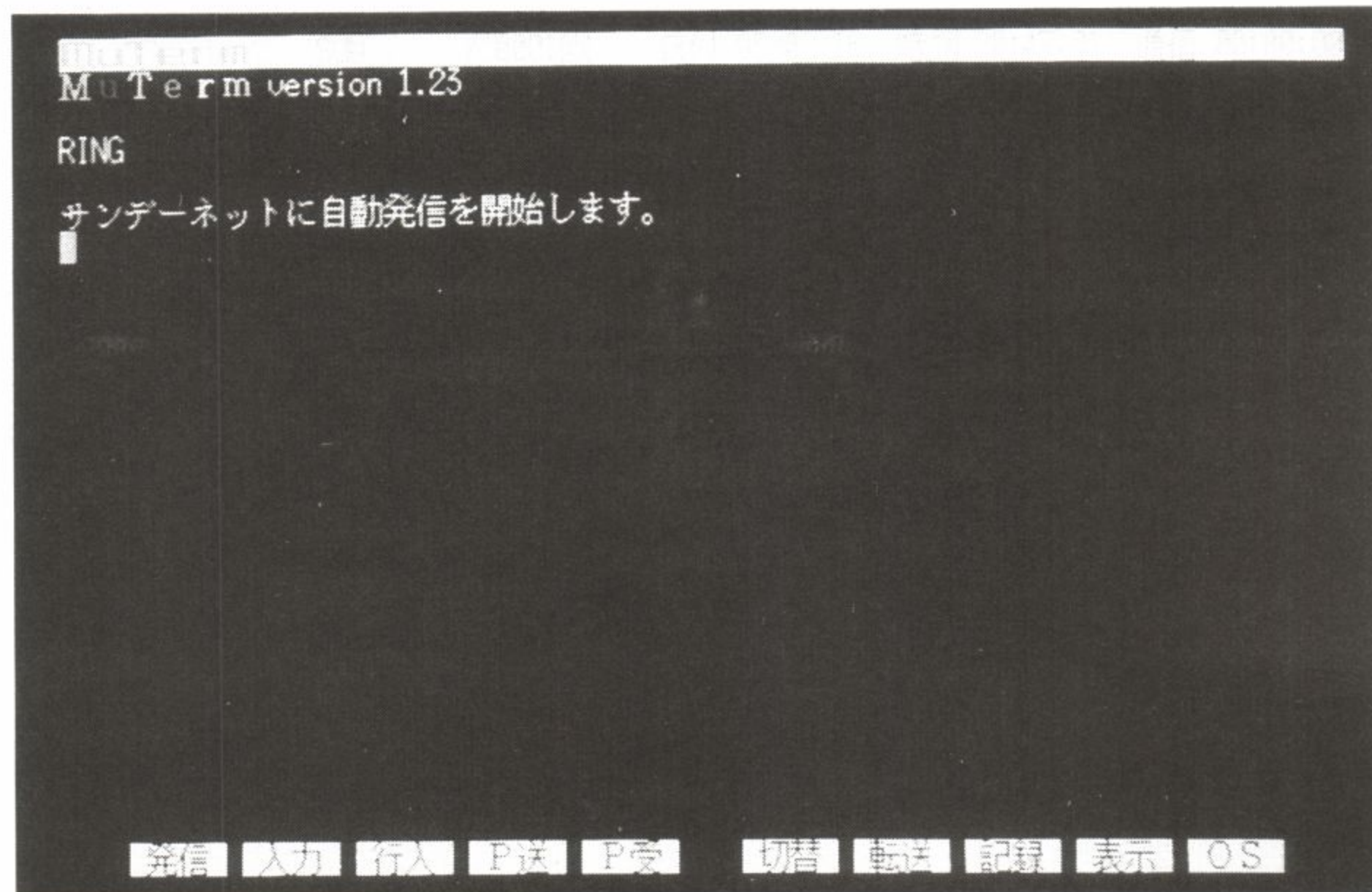
MuTermの起動方法はすでに説明しましたので、終了方法を説明しておきます。MuTermを終了させる場合は[SHIFT]+[BREAK]キーを押します。これでMuTermが終了できます。

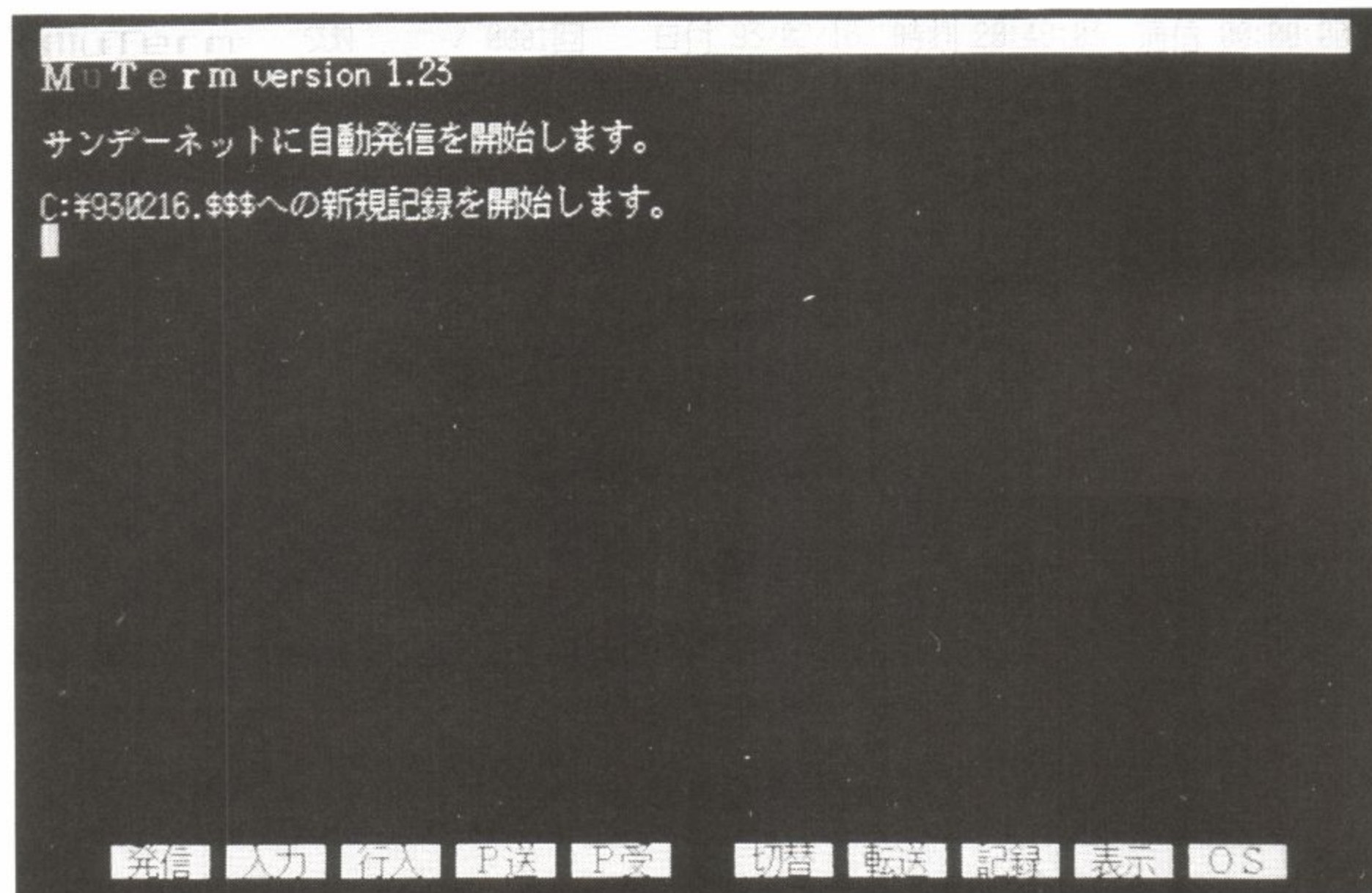
終了方法を確認したら、MuTermを再起動し、次にホストに電話をかけてみましょう。ここで、[F1]（「発信」）キーを押してください（Pht.2)。

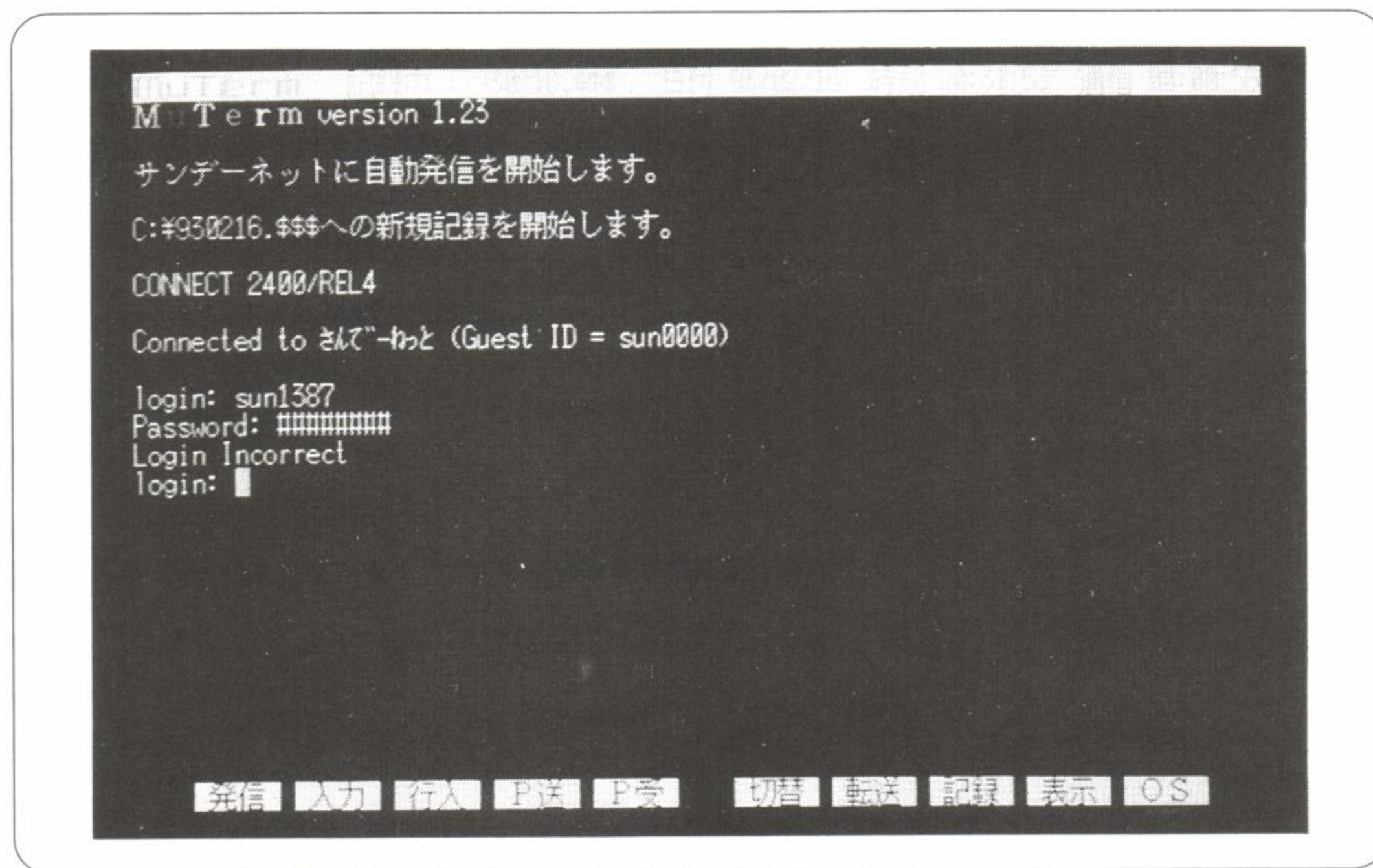
Pht.2 [F1]を押したときの画面

「ダイヤル発信先？」と聞いてきますが、ここではもう１度[CR]キーを押してください。すると、次のような画面になります（Pht.3)。

Pht.3 [F1]を押してから、[CR]を押したときの画面

ここで、「サンデーネット」にゲストログイン（ゲストでアクセス）するため、カーソルキーを「サンデーネット」にあわせ、[CR]キーを押してください（東京03区域外の方は、ATD03-3584-7791と、市外局番までダイヤルするようにMUTERM.CNFを書き換える

か、Pht.1の画面で「ATD0335847791」と手動でダイヤルしてください)。

Pht.4 MuTermがダイヤルをしているところ

モデムが電話をかけている間にログファイル(通信中の記録。画面に表示されたものがすべて記録される)をとるようにしておきます。MuTermは、デフォルト(初期設定)ではログをとるようになっていませんので、[F6](切替)キーを押してログをとるように設定しておきます(Pht.5)。

Pht.5 [F6]を押したときのメッセージ

サンデーネットの回線が空いていれば、コネクト（回線が接続）し、Pht.6のようにサンデーネットにログインします。

Pht.6　サンデーネットのタイトル画面

```
MuTerm version 1.23
サンデーネットに自動発信を開始します。
C:¥930216.$$$への新規記録を開始します。
CONNECT 2400/REL4
Connected to さんでーねっと (Guest ID = sun0000)
login: sun1387
Password: ##########
Login Incorrect
login:
発信 入力 行入 P送 P受 切替 転送 記録 表示 OS
```

ここでsun0000（サンデーネットの場合、ゲストログイン用のIDは、こう入力する。他のBBSの場合は、はじめてログインする場合は「GUEST」とか「TESTUSER」と入力することもある）と入力してください。

login: sun0000[CR]

続いて、

ハンドル名（ニックネーム）を入力してください　：

と聞いてくるので、ハンドル名（パソコン通信上で使う自分の呼び名。ログネームと呼ぶホスト局もある。本名でもかまわない）を入力してください。ここでは、「テスト」と入力します。

ここまで入力すると、下のような画面が表示されます。

Fig.1　サンデーネットのタイトル画面

```
☆ 通算ログイン回数　 : 546110回目
☆ 今回ログイン日時　 : 1992/11/07 13:53:41
この時間からの利用時間は 60 分間までです。

●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●

  テストさん，サンデーネットへようこそ！

  サンデーネットの秋期定期会員募集は締め切りました！　たくさんの応募，
  ありがとうございました．

  ＩＤはお申し込みから１ヶ月以内に電話連絡致しますので，留守がちな方
  は留守電のセットをお忘れなく！

  パスワードは大文字と小文字を区別しますので，ＩＤを受け取った方はご
  注意ください．

  サンデーネットのボードマップは MAP，オンラインマニュアルは MANU で
  ご覧になれます．

  表示の停止…CTRL+S，再開…CTRL+Q，中止…CTRL+C です．

●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●

        ＊＊　サンデーネット　Main Menu　＊＊

１．ｻﾝﾃﾞｰﾈｯﾄ・マップ(MAP)          ７．電子メール(MAIL)
２．ｻﾝﾃﾞｰﾈｯﾄ・事務局(SYS)          ８．端末設定(TERM)
３．電子掲示板(BBS)                 ９．プロフィール(PROF)
４．Ｍａｋｅｒ’ｓ Ｒｏｏｍ(MAKER)  10．チャット(CHAT)
５．ライブラリ(LIB)                 11．操作方法(MANU)
６．電子手帳(TECHO)                 12．終了(END)

MAIN (MAP, SYS, BBS, MAKER, LIB, TECHO, MAIL, TERM, PROF, CHAT, MANU, END, ?) :
```

これでログインが完了しました。

以下、MuTermのファンクションキーにそって、その機能を順に説明していきたいと思います。

### F1「発信」

まず、[F1]「発信」機能については、すでにサンデーネットへのログインの際に説明しました。おさらいをしておきましょう。

[F1]キーを押すと、呼び出すBBSの名前を入力するように聞いてきます。あらかじめ登録してあるBBSを呼び出す場合は[CR]キーを押し、BBS選択画面からカーソルキーの[↑]、[↓]でBBSを選びます。BBSを指定すると、そのデータがMUTERM.CNFに正しく設定されていれば、その選択したBBSに発信を行います(Pht.2)。

目的のネットに接続すると、接続したことを示すBEEP音が鳴ります。もしつながらなかった場合は、MuTermはリダイヤルを始めます。MuTermのリダイヤルには、次のような3つの種類があります。

1) 1ヵ所リダイヤル

１つのBBSに何度もリダイヤルします。MUTERM.CNFで１ヵ所しか登録していないか、オールダイヤルを選択してもセレクト状態の場所が１ヵ所しかない場合、このモードになります。サンデーネットにログインする際に使用したのは、このモードです。

2）連続ダイヤル

すべてのBBSに１回ずつリダイヤルします。BBS選択画面で一番上の“ALL”を指定したときに、セレクト状態のBBSが１ヵ所もなかった場合、このモードになります(Pht.7)。

3）BBSセレクトダイヤル

選択してあるBBSに上から順番にダイヤルします。複数のBBSを選択したあと、選択画面で一番上の“ALL”を指定したとき、このモードになります。

Pht.7　連続ダイヤルを選択している画面

### F2「入力」

「入力」とは、バックログの最後へエスケープシーケンスを含んだファイルを送り込み、「再生」機能（後述）で再度エスケープシーケンスを楽しむときなどに使用します。

どのように使うか、実際にやってみましょう。

まず、[F2]キーを押すと「入力ファイル名？」と聞いてきますので、ファイル名をフルパスで指定するか、そのまま[CR]キーを押します。[CR]キーを押すと、次のような画面が出てきます（Pht.8）。

Pht.8　起動時のカレントディレクトリ表示画面

＊＊　サンデーネット　Main Menu　＊＊

1. さんでーねっと・マップ(MAP)
2. さんでーねっと・事務局(SYS)
3. 電子掲示板(BBS)
4. Ｍａｋｅｒ'ｓ Ｒｏｏｍ(MAKER)
5. ライブラリ(LIB)
6. 電子手帳(TECHO)
7. 電子メール(MAIL)
8. 端末設定(TERM)
9. プロフィール(PROF)
10. チャット(CHAT)
11. 操作方法(MANU)
12. 終了(END)

MAIN (MAP,SYS,BBS,MAKER,LIB,TECHO,MAIL,TERM,PROF,CHAT,MANU,END,?) :

発信　入力　行入　P送　P受　切替　転送　記録　表示　OS

この画面に表示されているディレクトリは起動時のカレントディレクトリです。そのディレクトリに転送したいファイルがあったら[↓]キーで選択し、ない場合は[↑]キーでカーソルを“C:¥”（起動時のディレクトリが“C:¥”の場合）にあわせ、カーソルキーの[→][←]でドライブを変更し、ファイルを選択して[CR]キーを押すと、入力完了です。

ここでは、TEST.LOGという名前のファイルを選択することにします（Fig.2）。

Fig.2　TEST.LOGの内容

```
これは、MuTermのテキストアップテストです。

さささささささ颯爽と登場
もうだめやぁ

                                        テスト
```

このファイルを選択して[CR]キーを押すと、画面に「入力しました」と表示されます(Pht.9)。

Pht.9 入力完了

それでは、確認してみましょう。[UNDO]キーを押してみてください。バックログウィンドウが開きます(Pht.10)。

Pht.10 バックログウィンドウが開き、そこにFig.2のファイル内容が表示されている

うまく入力されているようですね。

### F3「行入」

「行入」とは1行入力のことです。[F3]キーを押すと行入力モードになり、入力した文字はファンクションキーの上の行に表示されます（Pht.11）。

Pht.11　[F3]を押したときの画面

適当な文字列を入力してください。たとえば、「おもちゃのちゃちゃちゃ」と入力したあと、[CR]キーを押します(Pht.12)。

Pht.12　行入力行に「おもちゃのちゃちゃちゃ」と入力されている画面

どうですか。うまく「おもちゃのちゃちゃちゃ」が送信されましたか？

### F4「P送」

「P送」とは、バイナリファイルを送信する機能です。MuTermから「XMODEM送信ファイル名」と送られてきたら、[F4]キーを押してください。[F4]キーを押すと、ファイル名を入力するように促されます(Pht.13)。

ここで[CR]キーを押すと、ファイル選択画面になり、カーソルキーの[↑][↓]で送信するファイルを選べます（一番上の起動したドライブ名を示すところにカーソルをあわせると、カーソルキーの[←][→]でドライブ選択が可能です)。

Pht.13 ファイル名入力を促している画面

[HELP]ウィンドウ、またはMUTREM.CNF内で送信プロトコルとしてXMODEMを選択している場合は、XMODEM（128バイトSUM)、XMODEM-CRC（128バイトCRC)、XMODEM-1K (1024バイトCRC) の3モードを自動的に判別し、使用できる中で最も効率のよいプロトコルでの送信を行います。また、YMODEMを選択している場合は、YMODEMとYMODEM-gを自動判別します。

ファイルを最後まで送信し終わると、BEEP音が鳴って終了を知らせます。

### F5「P受」

「P受」とは、XMODEM等のプロトコルを使った、バイナリデータの受信を行います。MuTermから「XMODEM受信ファイル名」と送られてきたら、[F5]キーを押してくだ

さい。[F5]キーを押すと、ファイル名（XMODEMの場合）またはパス名（YMODEMやM-LINKの場合）を入力するように促されます(Pht.14)。

**Pht.14** パス名入力を促している画面

ここで単に[CR]キーを押すと、YMODEMやM-LINKの場合ならカレントディレクトリを指定したことになり、バイナリデータの受信を開始します。XMODEMの場合は、キャンセルになります。

このうちXMODEMでは、[F4]「Ｐ送」と同様、３つのモードを自動的に判別し、受信します。最後まで受信できると、BEEP音が鳴って終了します。

**注意**

上記、[F4]「Ｐ送」、[F5]「Ｐ受」のバイナリファイル送受信は、サンデーネットではゲストには公開していない機能です。会員になったら使用できるようになりますので、注意してください。

### F6「切替」

「切替」とは、[F8]「記録」（後述）で記録していたものを、とりやめたり、再開したりするスイッチです。

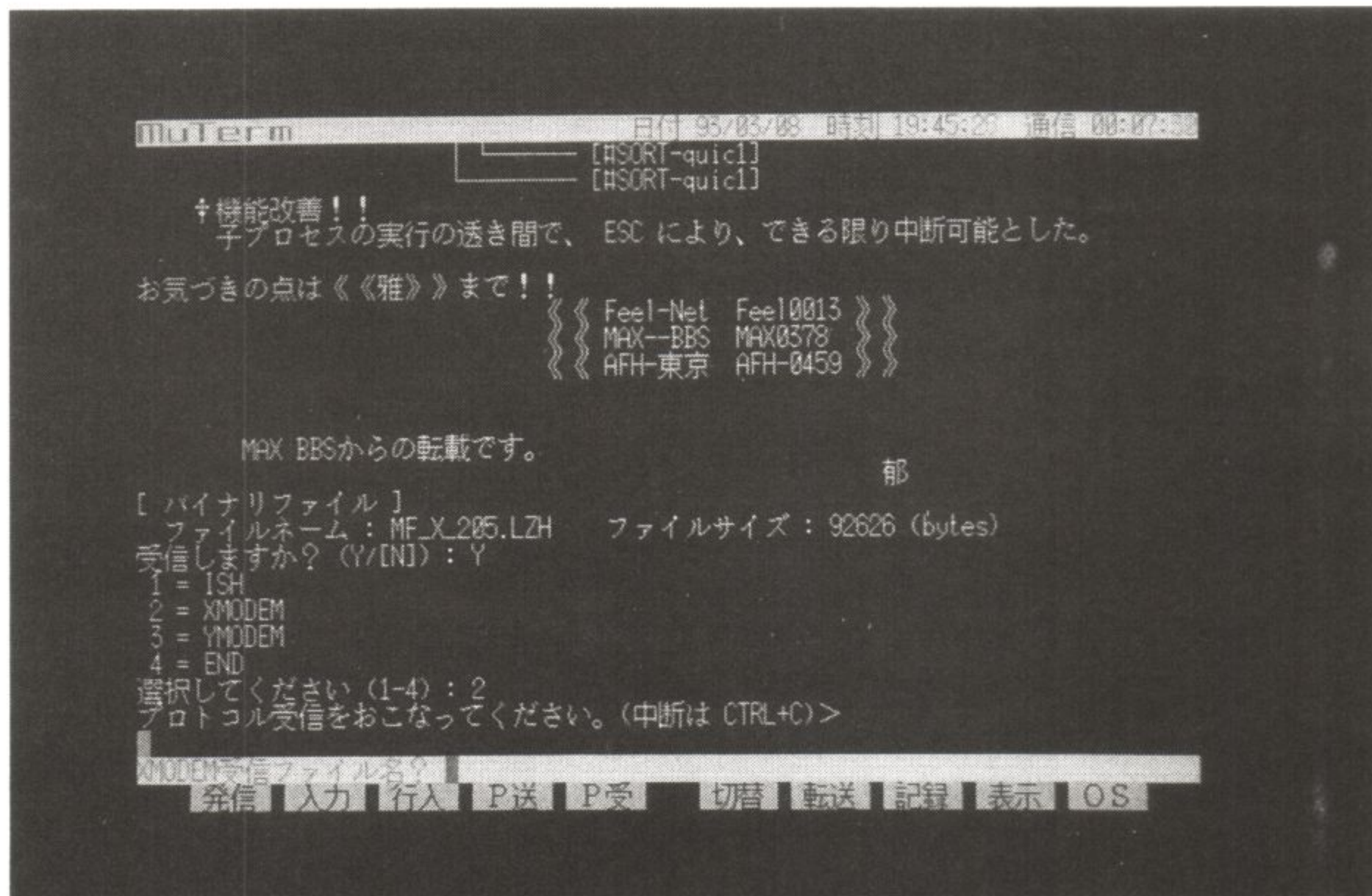
１度、[F6]キーを押してください（Pht.15)。

Pht.15　ログの記録を中断している画面

```
4006 sun3588 旅人(たびと) 93/02/14 01:22 Re4002:どもども(^^)            6
4005 sun0005 ゆうちゃん 93/02/14 00:10 Re4003:そういうものは             9
4004 sun3965 ＣＵＢ     93/02/13 23:10 あ、あ、あ、あ、                 10
4003 sun0851 一郎くん   93/02/13 12:05 Re4002:うう・・・                 7
4002 sun0005 ゆうちゃん 93/02/13 09:23 Re3999:はははははは・・・爆笑     12
4001 sun0047 まるみん☆ 93/02/13 09:07 Re4000:４０００番☆!!              9

Break..
BBS-TOWN-SRC (/3/1/2) (Write,Answer,List,Delete,Help,End) [4017] : /

      ＊＊ サンデーネット Main Menu ＊＊

1. さんでーねっと・マップ(MAP)        7. 電子メール(MAIL)
2. さんでーねっと・事務局(SYS)        8. 端末設定(TERM)
3. 電子掲示板(BBS)                    9. プロフィール(PROF)
4. Ｍａｋｅｒ'ｓ Ｒｏｏｍ(MAKER)     10. チャット(CHAT)
5. ライブラリ(LIB)                   11. 操作方法(MANU)
6. 電子手帳(TECHO)                   12. 終了(END)

MAIN (MAP,SYS,BBS,MAKER,LIB,TECHO,MAIL,TERM,PROF,CHAT,MANU,END,?) :
C:¥930216.$$$への記録を再開します。

C:¥930216.$$$への記録を中断します。

発信 入力 行入 Ｐ送 Ｐ受  切替 転送 記録 表示 ＯＳ
```

では、もう１度[F6]キーを押してください（Pht.16)。

Pht.16　ログの記録を再開している画面

```
ＭｕＴｅｒｍ version 1.23

RING

サンデーネットに自動発信を開始します。

C:¥930216.$$$への追加記録を開始します。

発信 入力 行入 Ｐ送 Ｐ受  切替 転送 記録 表示 ＯＳ
```

どうですか。このようにログファイル記録の中断／再開の切り替えができます。

### F7「転送」

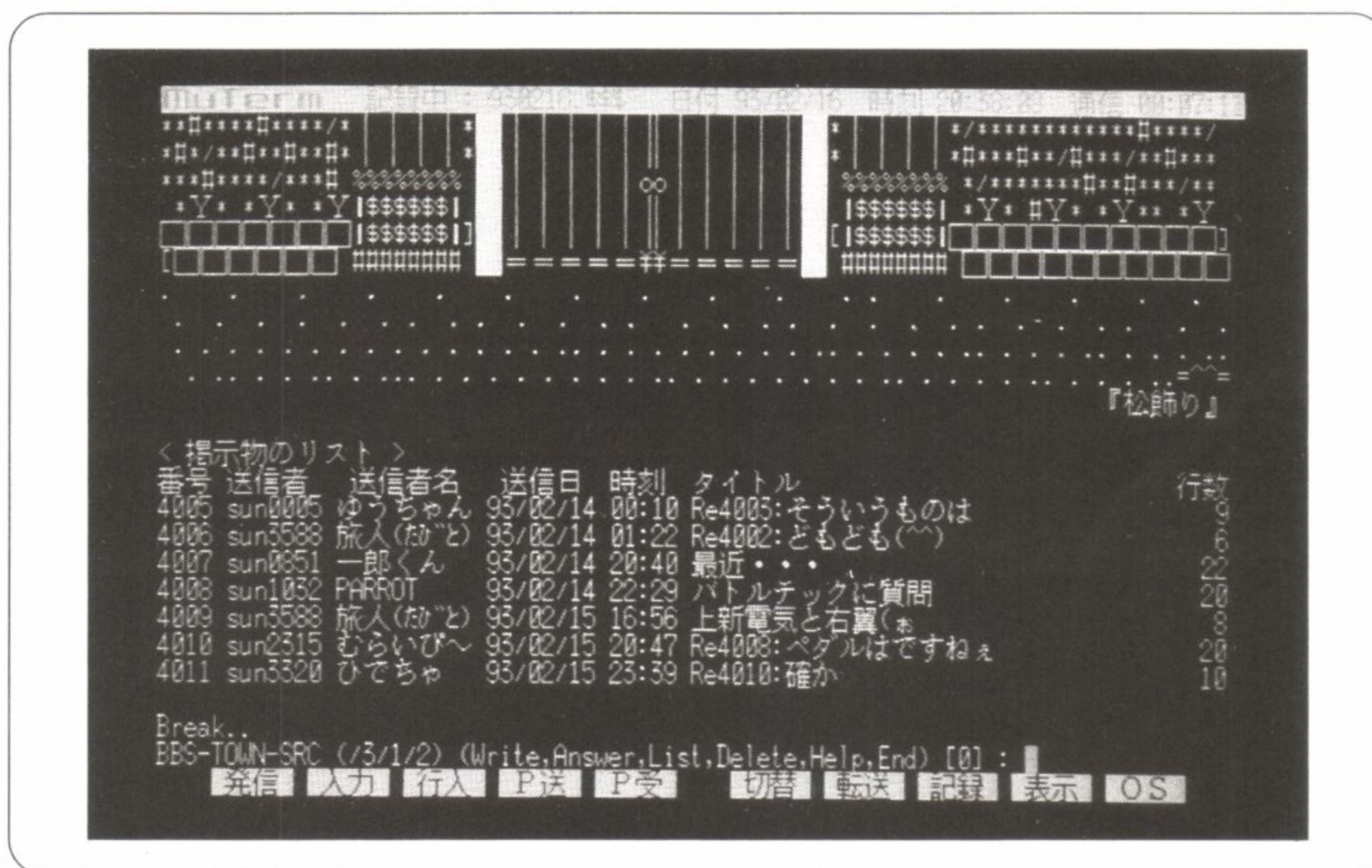
「転送」とは、ファイル送信をすることです。つまり、[F4]「Ｐ送」はプログラムなどの

送信ですが、「転送」は主に文章などのテキストファイルを送ることです。使用法は、[F2]の「入力」と同じく、[CR]キーを押すとファイラが起動し、カーソルキーで選択し、[CR]キーで決定すればファイル送信が行われます。

先ほどのTEST.LOG(Fig.2)をサンデーネットの落書き板(SRC)へ「転送」してみます。

まず、以下のようにメインメニューが表示されている状態で"SRC[CR]"とすれば、SRCボードへ移動します（Pht.17)。

MAIN（MAP,SYS,BBS,MAKER,LIB,TECHO,MAIL,TERM,PROF,CHAT,MANU,END,?）：

Pht.17　SRCボードのタイトル画面

ここで"W[CR]"と打ち込みます。ちなみに、この"W"はWRITEの頭文字です。

BBS-TOWN-SRC(/3/1/2)(Write,Answer,List,Delete,Help,End)[3753]：w[CR]

すると、次のように表示されますので、これから書き込む内容のタイトルをつけ、入力します（ここでは、「ごめんなさい」と入力します)。

< アーティクルの書き込み >

タイトル　：　ごめんなさい[CR]

“ごめんなさい[CR]”と打ち込んだら、次のような画面になります（Pht.18）。

Pht.18 書き込み画面

```
                                                           『松飾り』
< 掲示物のリスト >
番号 送信者  送信者名   送信日   時刻  タイトル                    行数
4005 sun0005 ゆうちゃん 93/02/14 00:10 Re4003:そういうものは          9
4006 sun3588 旅人(たびと) 93/02/14 01:22 Re4002:どもども(^^)        6
4007 sun0851 一郎くん   93/02/14 20:40 最近・・・                    22
4008 sun1052 PARROT     93/02/14 22:29 バトルチックに質問            20
4009 sun3588 旅人(たびと) 93/02/15 16:56 上新電気と右翼(ぉ           8
4010 sun2315 むらいぴ～ 93/02/15 20:47 Re4008:ペダルはですねぇ       20
4011 sun3320 ひでちゃ   93/02/15 23:39 Re4010:確か                  10

Break..
BBS-TOWN-SRC (/3/1/2) (Write,Answer,List,Delete,Help,End) [0] : W
< アーティクルの書き込み >
タイトル : ごめんなさい
文章の作成 ( 最大4000行 )
( 終了=.. 中止=.Q 表示=.D 修正=.En 削除=.Kn 挿入=.In 切替=.U 複写=.F )
  1:
 発信 入力 行入 P送 P受  切替 転送 記録 表示 OS
```

ここで[F7]キーを押し、“TEST.LOG”をカーソルキー[↓]で選択し、[CR]キーで決定してください。すると、ファイルが転送されます（Pht.19）。

Pht.19 TEST.LOG が転送された

```
4014 sun3720 E24GT      93/02/16 18:33 NSXだ!                        8
4015 sun1934 浅羽 広    93/02/16 20:42 Re4012:十人十色、千人千色です 14
もう新しいデータはありません。
BBS-TOWN-SRC (/3/1/2) (Write,Answer,List,Delete,Help,End) [0] : W
< アーティクルの書き込み >
タイトル : ごめんなさい
文章の作成 ( 最大4000行 )
( 終了=.. 中止=.Q 表示=.D 修正=.En 削除=.Kn 挿入=.In 切替=.U 複写=.F )
  1:
C:¥TEST.LOGから転送を開始します。
これは、MuTermのテキストアップテストです。
  2:
  3:
  4: さささささ颯爽と登場
  5:
  6: もうだめやぁ
  7:
  8:                                      テスト
  9:
 10:    とここまでがてすとでした
 11:
 12:    ごめんなさい
 13:
 発信 入力 行入 P送 P受  切替 転送 記録 表示 OS
```

ここまでの作業が終了したら、行頭に書き込みの終了を示す“.”を打ち込み、[CR]とし

ます。そして、サンデーネットへの「送信」を表す“1”を入力すれば、書き込みが終了します(Pht.20)。

Pht.20　“.[CR]”と打ち込み、書き込みを終了している画面

### F8「記録」

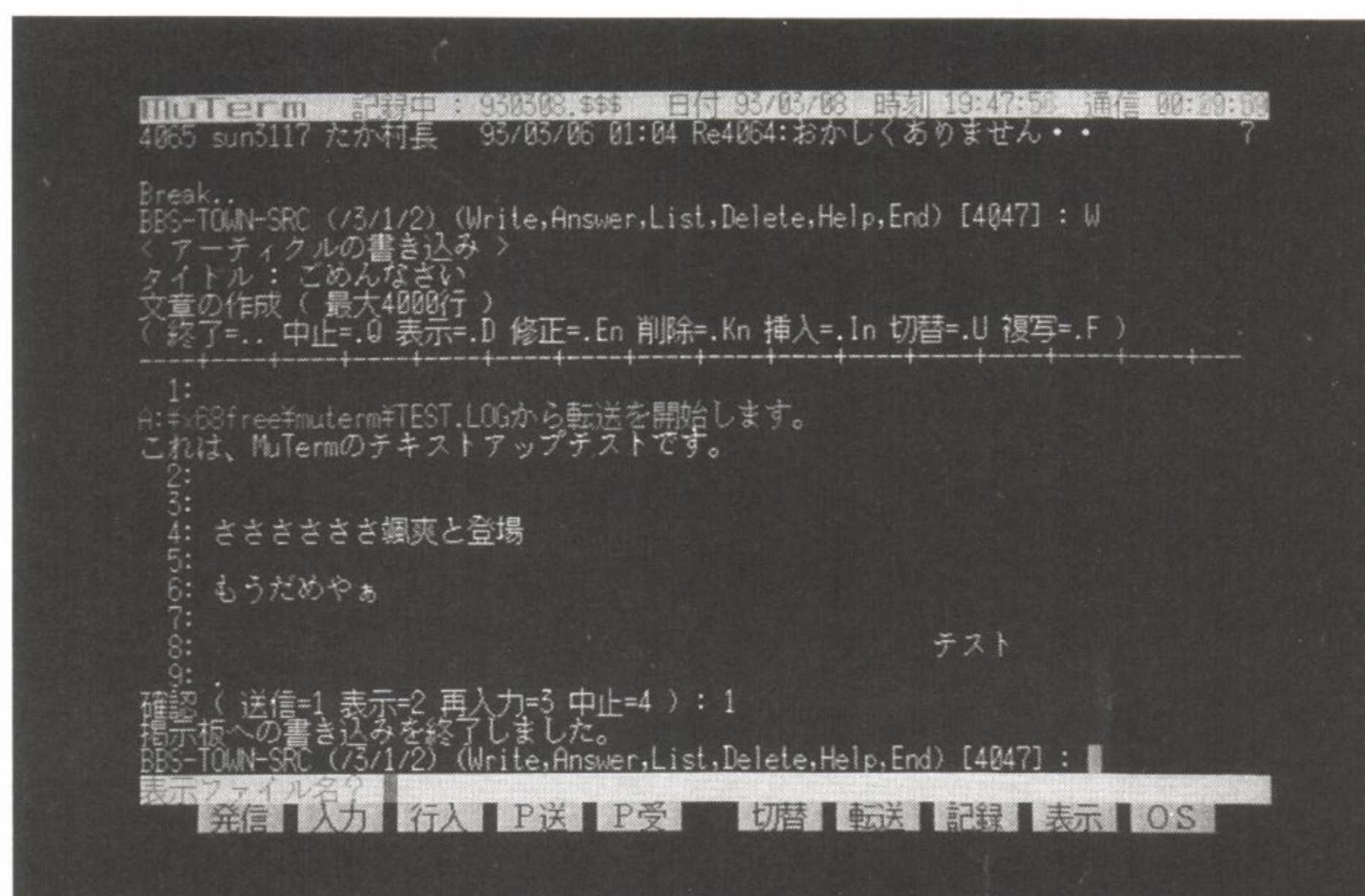
「記録」とは、ホスト側から送られてくる文字を画面に表示されたまま記録することを指します。[F8]キーを押すと、ファイル名を入力するように促してきます(Pht.21)。

Pht.21　ファイル名入力を促している画面

この機能では、既存のファイルと同じ名前で記録したい場合に限り、ファイル選択ウィンドウが利用できます（単に[CR]キーのみを押してください）。

### F9「表示」

「表示」とは、ファイル内容を画面表示させるための機能です。エスケープシーケンスをもう1度見たいときなどに使用します。使用法は、[F9]キーを押すとファイル名を入力するように促してきます(Pht.22)。表示したいファイルの名前を忘れてしまったとき、面倒なときは、単に[CR]キーのみを押してください。ファイル選択画面(Pht.8)になり、カーソルキーで表示するファイルを選べます。ただし、「表示」させた内容は、バックログには残りません。

Pht.22　ファイル名入力を促している画面

### F10「OS」

「OS」とは、チャイルドプロセスを利用したいときに使用します。

実行方法は、[F10]キーを押すと(Pht.23)、「OSコマンド？」と聞いてきますので、そのまま[CR]キーを押すと、COMMAND.Xが起動します。また、直接実行したいプログラムがあった場合はそのコマンドを入力してください。

Pht.23　「OSコマンド?」と入力を促している画面

```
muTerm  記録中 : 950508.$$$  日付 95/05/08  時刻 19:48:15  通信 00:10:19
4065 sun3117 たか村長   93/05/06 01:04 Re4064:おかしくありません・・    7

Break..
BBS-TOWN-SRC (/3/1/2) (Write,Answer,List,Delete,Help,End) [4047] : W
< アーティクルの書き込み >
タイトル : ごめんなさい
文章の作成 ( 最大4000行 )
( 終了=.. 中止=.Q 表示=.D 修正=.En 削除=.Kn 挿入=.In 切替=.U 複写=.F )
----+----+----+----+----+----+----+----+----+----+----+----+----+----
  1:
A:¥x68free¥muterm¥TEST.LOGから転送を開始します。
これは、MuTermのテキストアップテストです。
  2:
  3:
  4: さささささささ颯爽と登場
  5:
  6: もうだめやぁ
  7:
  8:                                          テスト
  9:
確認 ( 送信=1 表示=2 再入力=3 中止=4 ) : 1
掲示板への書き込みを終了しました。
BBS-TOWN-SRC (/3/1/2) (Write,Answer,List,Delete,Help,End) [4047] :
OSコマンド?
  発信 入力 行入 P送 P受   切替 転送 記録 表示 OS
```

これで、MuTermのファンクションキーに割り当てられた基本的な機能についての説明は終わりです。

### ●バックログウィンドウの操作

次に、バックログウィンドウが開かれているときの操作方法を説明します。

通常、通信画面では、さまざまなメッセージが送られてきます。そのメッセージはX68000の画面にすべて収まる場合もありますが、ときには行数が多すぎて画面の上のほうにスクロールして消えていってしまいます。このようなメッセージを、画面をさかのぼって読んだり、それらのメッセージを引用したり、保存したりする場合に必要なのがバックログの機能です。

バックログウィンドウのオープン、クローズは、[ROLL UP]、[ROLL DOWN]、[UNDO]キーのいずれかでできます。[ROLL UP]、[ROLL DOWN]キーではバックログの最新の位置から開きますが、[UNDO]キーでは前回にバックログウィンドウを閉じた画面から開かれます。

なお、以下にバックログウィンドウを開いた状態でのキー操作をまとめておきます。

### ●バックログウィンドウを開いた状態でのキー操作

| キー | 内　容 |
| --- | --- |
| [↑]（カーソル上） | バックログ表示を前方に1行だけスクロールさせる |
| [↓]（カーソル下） | バックログ表示を後方に1行だけスクロールさせる |
| [ROLL DOWN] | バックログ表示を前方に1画面分、スクロールさせる |

| | |
|---|---|
| [ROLL UP] | バックログ表示を後方に1画面分、スクロールさせる |
| [OPT.1](CTRL)+[ROLL DOWN] | バックログの表示位置を、最後にバックログウィンドウを開いた位置にする |
| [OPT.1](CTRL)+[ROLL UP] | バックログの表示位置を現在カーソルがある位置（最終位置）にする |
| [OPT.1](CTRL)+[↑] | 操作をするごとにバックログウィンドウが小さくなる。最小表示行数は2行 |
| [OPT.1](CTRL)+[↓] | 操作をするごとにバックログウィンドウが大きくなる。最大表示行数は21行 |
| [UNDO] | バックログウィンドウを閉じる（範囲指定は解除される） |

バックログウィンドウが開いているときはファンクションキーの働きが変わり、以下のような操作になります。ただし、リダイヤル中や転送・表示中、バイナリデータの送受信中は、これらのファンクションキー操作は無視されますので、注意してください。

以下に、その機能をまとめました。

### F1「前検」

「前検」とは、バックログバッファ内にある文字列を前方に検索し、その文字列がある行がバックログの一番上の行になるように移動します。文字列が見つからなかった場合はBEEP音を鳴らします。

### F2「後検」

「後検」とは、バックログバッファ内にある文字列を後方に検索し、その文字列がある行がバックログの一番上の行になるように移動します。文字列が見つからなかった場合はBEEP音を鳴らします。

### F3「行入」

「行入」とは、通常画面の「行入」と同じく、バックログをオープンしている間も同じように行入力の操作ができます。

### F4「前次」

「前次」とは、[F1]キーで指定した文字列を再び前方に検索し、その文字列のある行が一番上の行になるように移動します。また、何も指定していなかった場合は、「CONNECT」を検索します（続けて異なるBBSにアクセスしたときに、バックログの変わり目を見つける場合に便利です）。

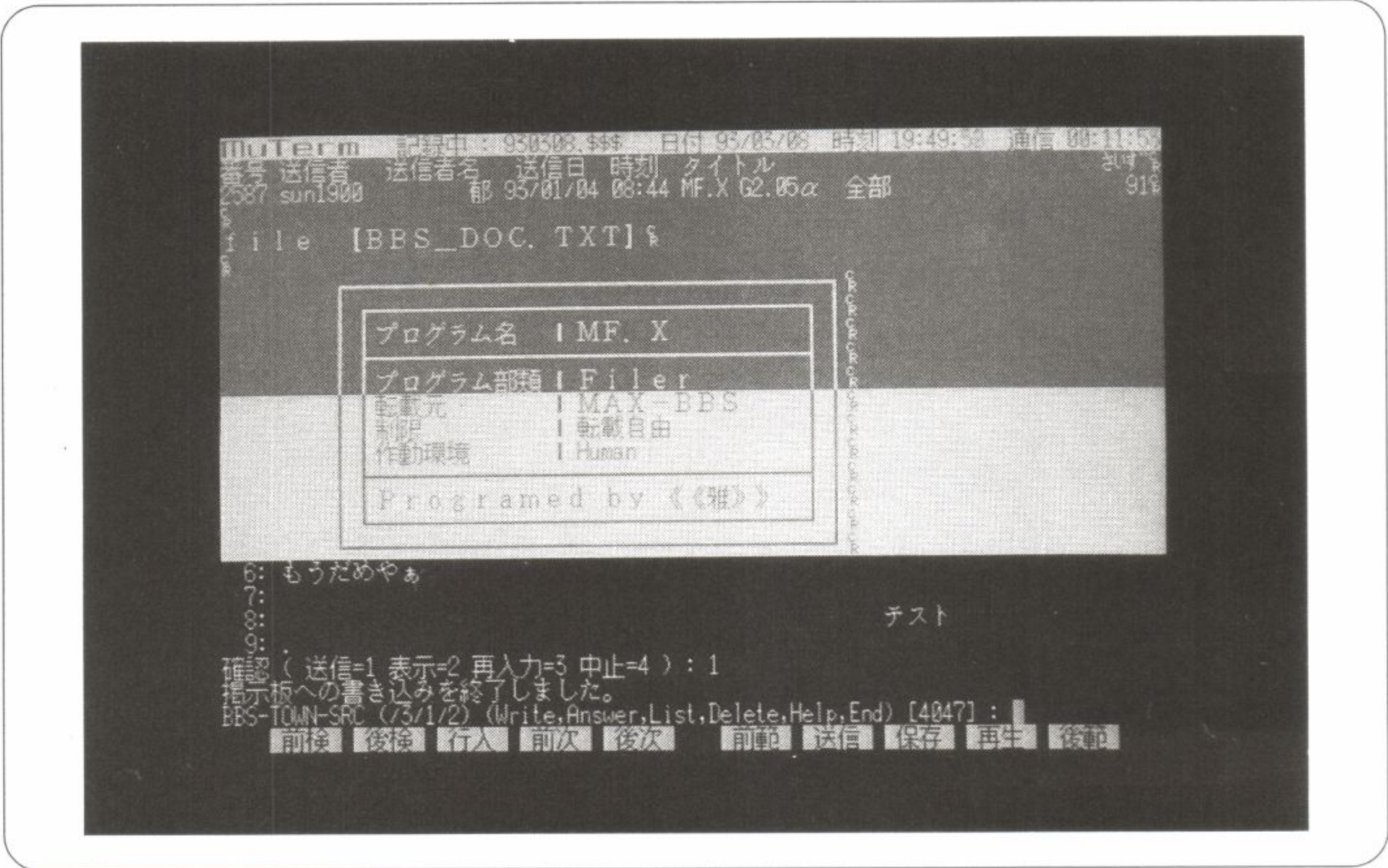
### F5「後次」

「後次」とは、[F2]キーで指定した文字列を再び後方に検索し、その文字列がある行が一

番上の行になるように移動します。また何も指定していなかった場合は、「CONNECT」を検索します（続けて異なるBBSにアクセスしたときに、バックログの変わり目を見つける場合に便利です）。

### F6「前範」

「前範」とは、バックログの内容を送信したり、保存したりする際の範囲を指定する機能です。[F6]キーを押すと、バックログの一番上のラインの色が変わります。カーソルキーや[ROLL UP]、[ROLL DOWN]キーで、色の違う部分の大きさを変えて範囲指定します。範囲指定中に[F6]キーをもう1度押すと、範囲指定を解除することができます(Pht.24)。

**Pht.24**　［前範］指定中の画面

以下の[F7]、[F8]、[F9]キーでは、この[F6]キーで範囲指定された部分を対象として作業します。指定を行っていない場合は、バックログウィンドウに現在表示されている部分を対象とします。

### F7「送信」

「送信」とは、範囲指定してある部分や、バックログ表示されている部分を入力として画面に出力することです。入力時に[CR]キーだけを押すと、範囲指定された部分を直接転送します。

### F8「保存」

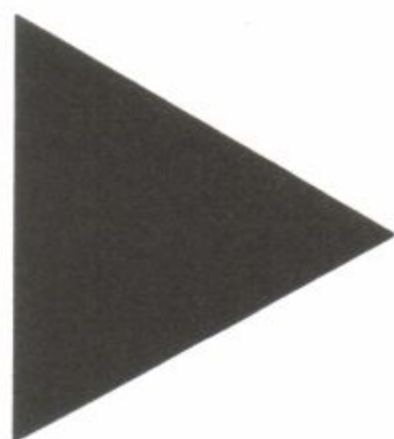
「保存」とは、範囲指定したバックログをファイルに保存します。その他の操作や注意事項などについては「記録」とほぼ同じですが、入力時にただ[CR]キーを押してもファイル選択ウィンドウは開きません。

### F9「再生」

「再生」とは、バックログ内の範囲指定した部分を直接表示します。この機能は、エスケープシーケンスをもう1度確認したいときなどに使用します。その他の操作や注意事項などについては「表示」とまったく同じです。

### F10「後範」バックログ後方範囲指定開始

「後範」とは、現在表示されているバックログの一番下の行から前（上）の行を「範囲指定」することです。[F10]キーを押すと、バックログの一番下のラインの色が変わります。カーソルキーや[ROLL UP]、[ROLL DOWN]キーで、色の違う部分の大きさを変えて範囲指定をします。範囲指定中に[F10]キーをもう1度押すと、範囲指定を解除することができます。

以上、MuTermの豊富な機能のうち、ファンクションキーに割り当てられた機能に絞って説明しました。これらの機能の他にも、MuTermにはまだまだたくさんの機能があります。しかし、その機能をすべてここで紹介することは紙数の都合でできません。詳しくは、MUTERM.MANをご覧ください。これらの基本的な機能を組み合わせることにより、より高度な使用法も利用できますので、各自研究してみてください。

# ish.x

## 通信必携のテキストーバイナリ変換プログラム

【概要】……………………　ish.xは、プログラムなどのバイナリファイルと、文字だけで構成されるテキストファイルの相互変換を行うプログラムです。

プログラムなどのバイナリファイルは、表示できる文字キャラクタ以外のコードを含んでいます。通常パソコン通信では、表示可能な文字しか取り扱うことができないので、バイナリファイルを転送するためには、XMODEMなどの転送用プロトコルをホスト局とパソコン側の双方で使用できることが必要です。

しかし、ish.xを用いることによって、送信側でバイナリファイルをテキストファイルに変換して転送し、受信側では転送されたテキストを逆変換してバイナリファイルに戻すことによって、バイナリ転送用プロトコルのサポートがされていないホスト局の場合でも、バイナリファイルの送受信ができます。

パソコン通信でテキストファイルの転送を行う場合、回線上のノイズなどによって、転送中の文字が別の文字に置き換わったり（文字化け）、文字が抜け落ちたり（文字落ち）する可能性があります。ish.xの場合、多少の文字化けや文字落ちがあってもきちんと元のファイルに逆変換できるように、変換されたテキストファイルにあらかじめ訂正用の情報を付加してあります。

なお、プログラム名のishは、オリジナルのMS-DOS版であるISH.COMの作者、石塚匡哉さんの名前に由来しています。

【作者】……………………☆紀

| | |
|---|---|
| PEKIN | hoshiki |
| 梁山泊 | hoshiki |
| NIFTY-Serve | NAH00103 |
| アスキーネット | PCS26838 |
| ゆいNET | yui00268 |

注：オリジナルのMS-DOS版は石塚氏作。本X68000版は、Ken、Gigo、Object.x各氏によるものにKzoo氏のUNIX版の一部を使用し、作者が追加修正したものです。

【作者からの言葉】……　もともとは単なるish.xユーザでした。多少気になることや不満があったものの、さしたる問題もなく使っていました。ところが、ある頃から原因不明の復元できないishファイルに出会うようになったのです。他に同様なことは聞いてないことと、復元できないと困るようなファイルでは起きたことがないことから、そのまま放置したままでいました。が、

あるとき、あることに気づきました。MS-DOSのISH ver. 2.0が関係していると。ish.xの原作者のKen、Gigo氏はもともとX68ユーザではないし、その後拡張したObject.x氏もX68関係の活動はほとんどされていないようだったし、MS-DOSのISH ver. 2の仕様と(8086のアセンブラだけど)ソースは公開されているし、ish.x関係のソースはすべて持ってるし、micの作者はver. 2対応はするつもりがないらしいと聞いたし……。なら、いろいろ直したいこともあったし、自分でやってみようかと手をつけたわけです。

目標は、まっとうなishファイルなら復元できるようにすること、自分に使いづらいところは改善すること、本家のISH.COMにあってish.xにない機能で気に入ったものを組み込むこと、細かいバグをとること、でした。おおむね実現したつもりですが、しかし、大きなバグを入れたまま現在に至ってます。mic.xを主に使う人が多いこととか、古いish.xでもさして不都合がないこととか、バグの修正をしなくてもほとんどクレームがなかったことも原因でしょう(^^;)。

さて、この本が出るまでに直るのやら……。そのバグとは……エラー訂正に失敗することです。MNP等を使用したモデムが当たり前で、文字化け等のエラーが起きることがあんまりないので、あまり実用上差し支えを感じていないために、つい放置してしまいました。いざとなれば、micも、古いish.xもありますから。でも、こうなったら、いいかげんに直さないと。最近クレームもきたことだし、ちょっと追加したい機能もあるし……。

**【使用法】**………………… ish [-オプションスイッチ] [ファイル名]

**【オプションスイッチ】**

**●共通のもの**

| オプションスイッチ | 機　能 | デフォルト |
|---|---|---|
| -f=[ディレクトリ名] | 指定したディレクトリに出力する | カレントディレクトリに出力 |
| -b=[バッファサイズ] | I/Oバッファのサイズを指定する | 32Kバイト |

**●復元時(テキストからバイナリへのデコード)に有効なもの**

| オプション | 機　能 | デフォルト |
|---|---|---|
| -a | すべてのファイルを復元する | (デフォルト) |
| -r | 同名のファイルがある場合、新しいファイルなら復元する | 拡張子を変更して復元 |
| -q | 同名のファイルがある場合は確認を求める | 拡張子を変更して復元 |
| -k | 不完全な復元ファイルを保存する | 不完全なファイルは削除 |

**●作成時(バイナリからテキストへのエンコード)に有効なもの**

| オプション | 機　能 | デフォルト |
|---|---|---|
| -ss | シフトJISコードで作成する | (復元する) |
| -s8 | 8bit JISコードで作成する | (復元する) |
| -s7 | 7bit JISコードで作成する | (復元する) |

| -sn | 半角カナ以外のシフトJISコードで作成する | (復元する) |
|---|---|---|
| -f | 標準出力に作成する | 拡張子を変更して作成 |
| -n | タイトルにエスケープシーケンスを含めない | (デフォルト) |
| -e | タイトルにエスケープシーケンスを含める | 含めない |
| -t[行数] | 指定行数ごとに行数表示を入れる | 50行ごとに入る |
| -t | 行数指定をしないと、行数表示も入らない | 50行ごとに入る |
| -om | 作成OSの種類をMS-DOS、Human68kにする | (デフォルト) |
| -ok | 作成OSの種類をOSK、OS-9拡張にする | MS-DOS、Human68k |
| -o9 | 作成OSの種類をOS-9にする | MS-DOS、Human68k |
| -oc | 作成OSの種類をCP/M、MSX-DOSにする | MS-DOS、Human68k |
| -oo | 作成OSの種類をその他のOSにする | MS-DOS、Human68k |
| -oa | 作成OSの種類をすべてのOSにする | MS-DOS、Human68k |

【主な機能】……………

## 1. ishテキストファイルの復元

ishで作成されたテキストを含むファイルの中から元のファイルを復元します。ishで作成されたテキストとは、たとえば、以下のようなテキストです。

```
<<< test.Lzh for Human68K ( use S-JIS ish ) [ 9 lines ] ish ver1.21 >>>
!!3D!!9Q!5"XRom7l6Kz!C$oXTRc!("H!&Y[mx!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!gRJt
!!3D!!9Q!5"XRom7l6Kz!C$oXTRc!("H!&Y[mx!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!gRJt
!!3D!!9Q!5"XRom7l6Kz!C$oXTRc!("H!&Y[mx!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!gRJt
%ﾓｵG1NP¥ay)ｳﾃn+ﾚsｴ9ﾒ臺WK帚>n癬<Zipl.ibｱ費ﾛ羚2'nﾔ-4gｼ中h､cHｴWﾑｱ､ｴEj|ﾊS>fo続,ｱ語ｴ
&b瓩hEﾂK+Vﾇn莫蹈ｫ|ﾆRweﾇzsﾇy唱ﾎｲﾉ'Lﾀ-霰QK問>n癬<EVC､R蘿F渠ﾝｷｽｧ軟ﾔ-¥s-PFﾛﾛ@@@]@}
｡D+4ｨ5ｪ覯ｭ[ｳpm兀ｱ|dW]ｵ]゛貰]y<ﾄ鏃wﾀﾎFｭg4ﾅﾈ楂ｮｳ[iaY|V贈ｵﾏ｢Yﾌ昿ｱA臑Fﾓﾝ､ｲﾖﾀｾE犇1B|
]Ks･¥7ﾛ孝Mｳ)n+杙2ﾙﾍ擦{}､Hｶ='=ﾍｶ_eﾅ霹去ir師ﾉ蒟sｻﾔ践i广|-゛｡嬌ｴ<ｬｳ蛤|夸+ﾈｮ*ｻ|@｡ｷﾋ|
--- test.Lzh (9/9) ---
```

通信のログファイルなどの場合、1つのファイルに複数のishテキストが含まれたり、ishテキストの中に無関係な文字が入ってしまっている場合もあります。しかし、復元時に気にする必要はほとんどありません。ishプログラムの側でishで作成されたテキスト部分を自動的に判別し、ほとんどの場合、問題なく復元を行います。

たとえば、ishテキストを含むファイルABC.LOGを復元する場合は、単に

ish ABC.LOG[CR]

とすればよいだけです。

すると、以下のようなメッセージが出力されてファイルが復元されます。

```
ish file converter for Human68K Ver1.21
Original idea By M.ishizuka(for MS-DOS)
Copyright 1988 - Pekin - (by Ken+Gigo)
Little modified by Obj.X Dec.22.1988
For N-JIS( from un*xish by kzoo ) & multi volume & modified by ☆紀 Apr.20.1991
<<< ish.doc from Human68K or MS-DOS ( use S-JIS ish ) 91/04/21 02:03:30 6622 Byt
es >>>
ish.doc : oo decoded.
LAP: 0.00 sec.
TOTAL: 0.00 seconds (process time) for decoding 1 file.
```

復元されるファイルと同名のファイルが復元先のディレクトリ（デフォルトではカレントディレクトリ。“-f=”オプションで指定可能）にすでに存在する場合は、復元するファイルの拡張子を“.000”に置き換えたファイル名（それもすでに存在する場合は“.001”、“.002”と順次変更していく）で復元します。なお、“-q”オプションをつけて実行すると、同名のファイルがある場合は上書きするかどうかの確認を求めるようになります。

復元の途中でishの復元能力を超えた文字化けなどが起きた場合、ishはその時点で復元をやめ、次のファイルの復元にかかります。復元に失敗したファイルは当然復元されません。“-k”オプションをつけて実行すると、復元できたところまでのファイルを出力します。

## 2. ishテキストファイルの作成

ファイルをishテキストに変更する場合は、“-s”オプションをつけてishを実行します。“-s”オプションは、使用する文字コードに応じて4種類ありますので、ホスト局が使用できるコードにあわせて使い分けてください。国内のパソコン通信ホスト局の場合は、通常シフトJISの漢字コードが使用できますから、“-ss”オプションを使用します。海外などで使用するときは“-s7”で使用するとよいでしょう。たとえば、test.lzhをishテキストに変換する場合は以下のようにします。

ish -ss test.lzh

すると、以下のようなメッセージを出力してishテキストファイルであるtest.ishが作成されます。

```
ish file converter for Human68K Ver1.21
Original idea By M.ishizuka(for MS-DOS)
Copyright 1988 - Pekin - (by Ken+Gigo)
Little modified by Obj.X Dec.22.1988
For N-JIS( from un*xish by kzoo ) & multi volume & modified by ☆紀 Apr.20.1991
Encoding "test.Lzh" to "test.ish" .. encoded.
LAP: 0.00 sec.
TOTAL: 1.00 seconds (process time) for encoding 1 file.
```

作成されたishテキストには、一定の行ごとと最後の行にそこまでの行数を示すフッタが書き込まれます。フッタが入る間隔はデフォルトでは50行ごとですが、"-t[行数]"オプションによって変更できます。また、"-e"オプションによって、最初のタイトルと各フッタ行にエスケープシーケンスで色をつけることも可能です。また、ishテキストのタイトル行には使用可能なOSが書き込まれます。通常はMS-DOSおよびHuman68kになっていますが、"-o"オプションで変更することが可能です。

**【その他】**…………………　X68000では、ish.xとほぼコンパチのプログラムとして、みるくさん作のmic.xというプログラムがあります。動作速度がish.xより高速であるなどの特徴があります。ただし、ファイル名8文字＋拡張子3文字を超える長いファイル名を独自の方法で格納保存していること（ish.xでは、基本的にファイル名8文字＋拡張子3文字しか認識しない）や、0x0d([CTRL]-[M])のみでは行末と認識しないことなど、微妙な点で動作に違いがありますので注意が必要です。詳しくはドキュメントを参照してください。

# LHA.x

## 高率圧縮を行う標準アーカイバソフト

【概要】……………………　LHA.xは、複数のファイルを取りまとめ、圧縮するためのプログラムです。通常、フリーソフトウェアとして配布されているプログラムは、プログラム本体の他にドキュメントや環境設定ファイル、ソースファイルなど、複数のファイルがいっしょになっています。これら複数のファイルを1つのファイルにまとめたり、逆にまとめたファイルから元の複数のファイルを復元したりするソフトウェアを「アーカイバ（書庫管理）ソフト」といいます。このアーカイバを用いることによって、必要なファイルが散逸することを防ぎ、プログラムの取り扱いを容易にすることができます。

最近のアーカイバソフトには、取りまとめと同時に、まとめたファイルの圧縮を行うものが多く存在します。圧縮を行うことによって、保存用のディスクスペースの節約や、通信によるプログラム転送の場合の送受信時間の短縮（つまり、コスト削減）を行うことができます。

LHAは、これらの圧縮機能付きアーカイバソフトの中でも、とりわけ高い圧縮率を誇るソフトウェアの1つです。圧縮するファイルの種類にもよりますが、一般的に規則性が高いテキストファイルなどの場合は2分の1以下のサイズになることが大部分です。LHAは、その旧版であるLHarcのMS-DOSバージョンが発表された当時から、その高圧縮率で瞬く間に広まり、現在では事実上の業界標準の圧縮プログラムとなっています。パソコン通信を通して海外にも紹介され、日本のみならず世界中で広く使用されている数少ない日本製ソフトウェアの1つにもなっています。

LHA.xは、このLHAのX68000版です。LHA.xはオリジナルの機能に加えて、MS-DOSより長いHuman68kのファイル名への対応や空ディレクトリの圧縮機能、さらにはTwentyOneなどの他の便利なフリーソフトウェアへの対応やシンボリックリンク機能への対応など、独自のさまざまな機能拡張をほどこした、大変便利なプログラムとなっています。

【作者】…………………　岡田紀雄　　NIFTY-Serve　TBE01054（TBE01054@niftyserve.or.jp）
ともちゃんNET　1

注：オリジナルのMS-DOS版作者は吉崎氏です。

【作者からの言葉】……　LHAの前身であるLHarcの魅力にひかれて、その圧縮アルゴリズムを知るため、LHarcのソースを探し求めていたところ、LHxなる次世代LHarcのソースを見つけまし

た。幸いなことに、それは、C言語で記述されていたため、DOS固有の命令、CPU(MPU)の相違に注意することにより、多少のバグは残っていましたが、移植に成功しました。このとき、たくさんの方々からバグ情報をいただけたおかげで、今日のLHA for X68000があるものと感謝しております。また、数々の拡張に対しアドバイスしていただいたことにも感謝しております。そして、なによりもこの素晴らしいアーカイバのソースを快く提供してくださった吉崎氏に深く感謝の意を表します。

ところで、このLHAですが、本家LHA for MS-DOSでは、-lh6- なる、さらなる圧縮比向上に向けての改良がなされ、現在試作段階にあるわけですが、X68000版もテストバージョンとして稼動しております。テスト段階ということで発表を控えて欲しいとのことですので、今回のディスクには入っていませんが、いつかまたどこかで……。

最後に一言。HDDなどの大容量のバックアップツールが乏しい中、そのかわりにLHAをなんとか工夫して使っていただいている方もおられるようです。おおいに結構なことなのですが、なにぶんLHAはどんなファイルに対しても、という具合にかなりの汎用性を持たせて作られたものですので、機能的に十分とはいえません。そんな中、各種サポートツールが開発されていますので、それらとあわせて使われることを望みます。X68000ならではの贅沢な環境ともいえるでしょう。

**【推薦します】**…………　フリーソフトの中では、1、2を争うほどの有名なプログラム。ネットにアップロードされているプログラムやデータのほとんどがこれを使ってアーカイブされているほど。ディレクトリごと圧縮したり、自己解凍ファイルを作ったりすることもできるなど多機能で、圧縮率もかなり高い。まず最初に入手すべきフリーソフトウェア。

MAX BBS Vorspiel

**【使用法】**……………　LHA　［コマンド名］［-オプション］［アーカイブ(書庫)ファイル名］［ファイル名　...］

**【コマンド】**…………　**コマンド**(LHAの直後に記述し、全体の処理内容を決定するもの。デフォルトは“l(エル)”コマンドに設定されています)

| コマンド | 機　能 |
|---|---|
| a | ファイルを凍結（圧縮）し、アーカイブファイルに追加、置き換えを行う（同名ファイルがあっても行う） |
| u | ファイルを凍結し、書庫内に同名ファイルがある場合はタイムスタンプの新しいものに置き換わる（ない場合は追加） |
| m | ファイルを凍結し、書庫内に同名ファイルがある場合はタイムスタンプの新しいものに置き換わる（ない場合は追加。元ファイルは削除される） |
| f | ファイルを凍結し、書庫内のファイルの内容は無視して新しいものに置き換わる |
| d | 書庫内のファイルを削除する |
| e | 書庫ファイルからファイルを解凍（復元）する |
| x | 書庫ファイルからファイルをサブディレクトリまで含め解凍（復元）する |
| p | 書庫内のファイルを表示する |

| | |
|---|---|
| l | 書庫内のファイルの一覧を表示する |
| v | 書庫内のファイルの一覧をディレクトリまで含め表示する |
| t | 書庫ファイルを検査する |
| s | 自己解凍形式の書庫ファイルを生成する |

**【オプション】**………… **オプション**（コマンドのあとに“-”付きで記述し、細かい動作を制御するもの）

| オプション | 機　能 | デフォルト |
|---|---|---|
| -x? | ディレクトリ名の有効・付加 | |
| | -x0　：無効・付加しない | e コマンド時 |
| | -x1　：有効・付加する | x コマンド時 |
| -p? | ファイル名検索厳密化 | TwentyOne次第 |
| | -p0　：厳密に行わない | |
| | -p1　：厳密に行う | |
| -c? | 日付の照合 | |
| | -c0　：日付の照合を行う | a コマンド以外 |
| | -c1　：日付の照合を行わない | a コマンド時 |
| -m? | 問い合わせ（Yes/No） | |
| | -m0　：行う | a コマンド以外 |
| | -m1　：行わない(重ね書き) | a コマンド時 |
| | -m2　：行わない（名前連番変更） | |
| -a? | ファイル属性 | |
| | -a0　：ファイル属性を無視・保存しない | e コマンド時 |
| | -a1　：ファイル属性を有効・保存する | x コマンド時 |
| -r? | 凍結時のサブディレクトリ処理 | -r0 |
| | -r0　：ディレクトリは対象外とする | |
| | -r1　：子ディレクトリまで対象とする | |
| | -r2　：子・孫……再帰的に処理する | |
| -w? | 作業ディレクトリ | -w1 |
| | -w0　：作業ディレクトリを使用しない | |
| | -w1　：環境変数 temp の指定による | |
| | -wXXX　：ディレクトリを設定し、それとする | |
| -n? | 表示形式を変更する | -n0 |
| | -n0　：スタンダード（横96文字） | |
| | -n1～3　：実際に使ってみてください | |
| -t? | 書庫の更新日時 | -t0 |
| | -t0　：(再)凍結作業時刻にする | |
| | -t1　：凍結対象ファイルの最新日付にする | |
| -z? | ファイルの圧縮格納 | -z0 |
| | -z0　：全ファイルを圧縮対象とする | |
| | -z1　：圧縮せずに凍結する | |
| | -z2　：環境変数 LHA_NON_SUF により非圧縮ファイル拡張子の設定を行う | |
| -o? | ファイルの圧縮形式 | -o0 |
| | -o0　：-lh4-, -lh5- 圧縮 | |
| | -o1　：-lh1-（LHarc互換） | |
| -h? | ヘッダの形式 | -h1 |
| | -h0　：レベル 0（LHarc互換のヘッダ） | |
| | -h1　：レベル 1（スタンダード） | |
| | -h2　：レベル 2（拡張形式） | |
| -i? | ファイル名の大文字・小文字識別 | TwentyOne次第 |
| | -i0　：識別しない | |
| | -i1　：識別する | |

| -e? | 空ディレクトリ書庫<br>-e0　：-lhd- として登録<br>-e1　：凍結対象としない | -e0 |
|---|---|---|

【主な機能】……………

## 1. 環境変数

LHA.xは、以下の環境変数を参照します。必要に応じて設定してください。

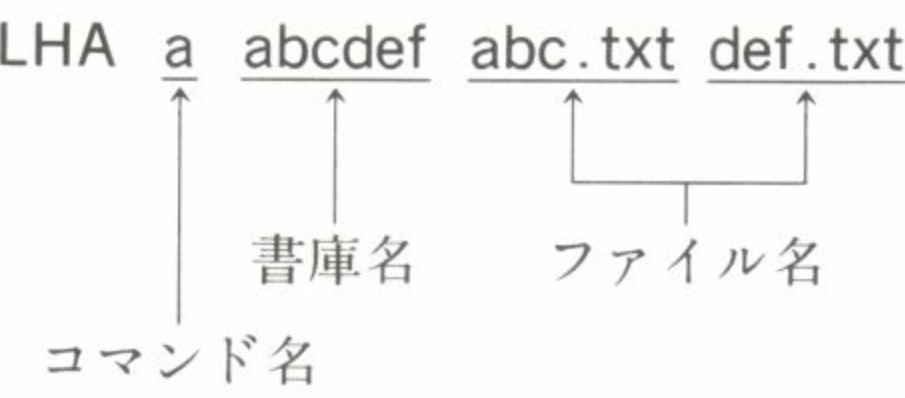
| 環境変数名 | 内　容 |
|---|---|
| LHA | デフォルトのオプションを設定する<br>いつも指定するオプションを設定しておくと便利 |
| temp | 作業ファイルを作成するディレクトリを設定する。"-w?"オプションと同じ。RAMディスクなどの高速なメディアに指定したほうがよい |
| LHA_CMD | コマンドを省略した場合の動作を設定する<br>設定がないと"l"コマンドとして動作する |
| LHA_NON_SUF | "-z2"オプション使用時に圧縮しないファイルの拡張子を指定する<br>圧縮がきかないタイプのファイル（すでに圧縮されているファイルなど）の拡張子を指定しておくことによって、"-z2"オプションで圧縮時間を短縮することができる |
| PAGER | "p"コマンドでファイルを表示するとき、ファイルを引き渡すコマンドを指定する<br>設定がない場合は、"p"コマンドはHuman68kのTYPEコマンドのように動作する |

設定はHuman68kのSETコマンドで行います（例：SET LHA = -z2)。SETコマンドについては、Human68kの「ユーザーズマニュアル」を参照してください。

## 2. ファイルの凍結（圧縮）

LHAでは、ファイルを取りまとめて圧縮することを「凍結する」といっています。このためのコマンドが"a"コマンドです。たとえば、ファイルabc.txtとdef.txtを、abcdef.Lzhという書庫ファイル名(ファイルをとりまとめする際に、ユーザがつけるファイル名)で凍結するには以下のようにします。

```
LHA a abcdef abc.txt def.txt
```

a：コマンド名
abcdef：書庫名
abc.txt def.txt：ファイル名

すると、以下のようなメッセージを表示してabc.txt、def.txtの圧縮が行われ、abcdef.Lzhが作成されます。abcdef.Lzhがすでに存在する場合は、そのファイルの中のabc.txtやdef.txtが更新されます。

```
Creating Archive : abcdef.Lzh

abc.txt              : 圧縮終了   [    13772 ] (     6051 ) 圧縮比 :  44%
def.txt              : 圧縮終了   [     2269 ] (     1189 ) 圧縮比 :  52%

Copying Temp to .Lzh ••• Done.
```

この例の場合は、以下のように、圧縮されたファイルのサイズは元ファイルの半分以下になりました。

| | | | | |
|---|---|---|---|---|
| abc | txt | 13772 | 92-05-31 | 2：06：00 |
| def | txt | 2269 | 92-05-31 | 2：06：00 |
| abcdef | Lzh | 7309 | 93-02-09 | 8：20：14 |

アーカイブファイルの拡張子は、デフォルトで.Lzhです。変えたい場合は拡張子まで指定すれば可能です（しかし、混乱を避けるためにもLHAのアーカイブファイルの拡張子は.Lzhにしておいたほうがいいでしょう)。添付のバイナリ差分ファイルをLHA.xに適用すると（やり方はBdif/Bupの説明ページを参照してください)、デフォルトの拡張子がMS-DOS版と同様の大文字の.LZHになります。MS-DOSマシンとファイルのやりとりを行う場合はこちらが便利でしょう。

“a”コマンドでは、アーカイブファイルの中に圧縮するファイルと同名のファイルが存在すると無条件で上書きします。“u”コマンドを使用すると、ファイルのタイムスタンプを比較して、日時の新しいほうのファイルを凍結します。

圧縮は、通常指定したファイルについてのみ行います。ディレクトリ構造をまるごと圧縮するためには、“-x”、“-r2”、“-a”などのオプションを同時に指定します。たとえば、SYSディレクトリ以下のファイルをまるごとdirtree.Lzhに凍結するには、以下のように記述します。ここで各オプションは、数字1がついている場合と同様の意味です（つまり、-xは-x1と同様ということです)。

LHA a -x -r2 -a dirtree SYS/＊.＊[CR]

また、次のような書き方もできます。

LHA a -xr2a dirtree SYS/＊.＊

つまり、オプションは続けて書く場合に限って“-”を省略できます。

### 3. 凍結ファイルの削除

アーカイブファイル内から凍結されたファイルの一部を削除するためには“d”コマンド

を使用します。たとえば、先ほどのアーカイブファイルからdef.txtを削除するには、以下のように指定します。

LHA d abcdef def.txt[CR]

すると、以下のようなメッセージを出力し、ファイルが書庫から削除されます。

```
Deleting from Archive : abcdef.Lzh

Deleted def.txt

Copying Temp to .Lzh ··· Done.
```

## 4．ファイルの解凍（復元）

圧縮されたアーカイブファイルから元のファイルに復元することを、LHAでは「解凍する」といいます。解凍のためには“e”コマンドと“x”コマンドがありますが、サブディレクトリを含めて解凍する場合は、通常“x”コマンドを使用します。最初に圧縮したabcdef.Lzhを解凍する場合は、以下のように記述します。

LHA x abcdef[CR]

すると、以下のメッセージを出力し、書庫が解凍されます。

```
Extracting from Archive : abcdef.Lzh

abc.txt                 : 解凍終了    [    13772 ] (    13772 )
def.txt                 : 解凍終了    [     2269 ] (     2269 )
```

なお、アーカイブファイルの中から特定のファイルだけを解凍する場合は、書庫ファイル名に続けて取り出したいファイル名を指定すれば、そのファイルだけを解凍することもできます。

## 5．アーカイブファイルのファイル一覧

書庫内のファイル一覧を表示するには“l(エル)”コマンドを使用します。たとえば、先ほどのabcdef.Lzhの場合は以下のようになります。

LHA l abcdef[CR]

出力は以下のとおりです。表示形式は“-x”や“-n”などのオプションで変更することができます。以下の例は、“-n1”オプションで横80桁に出力したものです

```
 ファイル名      日付      時刻      サイズ      書庫     圧縮比  属性  形式  CRC
============== ======== ======== ========== ========== ======== ==== ===== ====
<< dir >> / (root)
 abc.txt       92/05/31 02:06:00      13772       6051 ( 43.9%) -arw -lh5- 6F51
 def.txt       92/05/31 02:06:00       2269       1189 ( 52.4%) -arw -lh5- EE13
============== ======== ======== ========== ========== ======== ==== ===== ====
       2 files 93/02/09 08:20:14      16041       7240 ( 45.1%)
```

**【その他】**……………… LHA.xに似た機能を持つものとしては、みるくさん作のxx.xなどがあります。こちらは解凍専用で、LHAで圧縮されたファイルを高速に解凍します(ただし、ディレクトリのアーカイブなどには対応していません)。

アーカイバソフトとしては、他にもLHarcからX68000に移植されたlh.x(やーさん/まりこさん作)もあります。最近の版では、LHAと同様の圧縮形式にも対応しています。また、LHA(LHarc)登場以前の古いアーカイブファイルには、拡張子が“.ZOO”という形式のものも存在します。これは、ZOO.X（Rahul Dhesiさん作/atnaSさん移植）で圧縮されたアーカイブファイルです。また、最近MS-DOSマシンでは、LHAと同等レベルの圧縮率で、より高速な圧縮・復元を行うPKZIPが広まっていますが、X68000では先ほど紹介したxx.xが解凍用として使用できます（一部のフォーマットには未対応)。

# Bdif.x & Bup.x

## 通信ユーザ必携のバイナリ差分抽出更新プログラム

【概要】……………………　Bdif.x & Bup.xは、主にプログラム等のバイナリデータの差分情報を抽出したり、その差分情報をもとにしてデータの更新を行うプログラムです。Bdif.xにより差分情報を抽出して、Bup.xによってデータを更新します。

具体的にどういう局面で使うかを説明しましょう。たとえば、GNU（第1章参照）関係等の巨大なプログラムのバージョンアップが行われたとします。これをネットからダウンロードしようとすると、ダウンロードするだけで電話代も相当かかってしまいます。そのような場合、変更された部分だけ取り出せれば、時間もお金もかからずにすみます。つまり、Bdif.xは、その差分だけを抽出するプログラムなのです。こういった事情は、短い間隔で次々にバージョンアップが行われるフリーソフトウェアの場合についてもいえます。Bdif.xおよびBup.xはパソコン通信をしている人、必携のプログラムといえます。

【作者】……………………

| ひがしで | MAX BBS | 10 |
|---|---|---|
| | 梁山泊 | HIGASHID |
| | ひるま-ねっと | 15 |

【作者からの言葉】……［動機］

某80x86機にはbdiff.exe、ldiff.exeと似たようなツールがあるのにX68にはないのです。ある極悪人に「DOSにはこんなのがあるんだけど、X68にもあるといいね」とそそのかされ、気がつくとできあがってました。

［速度］

10MHzのMC68000には荷が重いらしく、遅いです。最悪でも寝る前に作業を始めれば起きる頃には終わっているでしょう。なお、デフォルトのオプションは、LHAをかけたあとのサイズが小さくなるように指定されています。

［注意］

Bdif.xとbdiff.exe、ldiff.exeは目的は同じですが、互換性はまったくありません（本当はbdiff.exeと互換を取りたかったのですが、連絡が取れなかったので）。

【操作方法】………………　操作方法は、いたって簡単です。

新・旧ファイルの差分をとるには、

Bdif　旧ファイル名　新ファイル名[CR]

　とします。これで旧ファイル名の拡張子を“bfd”に変えた新ファイル名.bfdという差分ファイルが作成されます。旧ファイルとこの差分ファイルから新ファイルにバージョンアップするには、

Bup　新ファイル名.bfd[CR]

とします。これで 新ファイルが作成されます。
　たとえば、手元のプログラム “foo.x” (バージョン 1.00 とします) を改良してバージョン 1.01 のものを作る場合、

A:¥NEW¥foo.x　　←新バージョン
A:¥OLD¥foo.x　　←旧バージョン
Bdif A:¥OLD¥foo.x A:¥NEW¥foo.x [CR]

と実行することにより、“foo.bfd”という差分ファイルが作成されます。これを差分ファイルとしてパソコン通信などで配布すればよいのです。旧バージョンを使っている人は、この差分ファイルを入手し、“foo.bfd”と“foo.x” (バージョン 1.00) を同じディレクトリに置いて、

Bup foo.bfd [CR]

と実行することにより、“foo.x” (バージョン 1.01) が作られます。
　Bdif.xの差分抽出については、元のファイルがバイナリデータの場合は、ほとんどオプションをつける必要はありませんが、テキストデータの場合は、“-D” をつけてください。これは、通常プログラム等では2バイトごとにペアとなっている (MPU68000シリーズでは、実行プログラムで奇数バイトから命令が始まることが許されていないため) ものが、テキストでは1バイトごとに変更されていることが多いので、1バイトごとの差分を取るオプションをつけているのです。

【オプション】…………

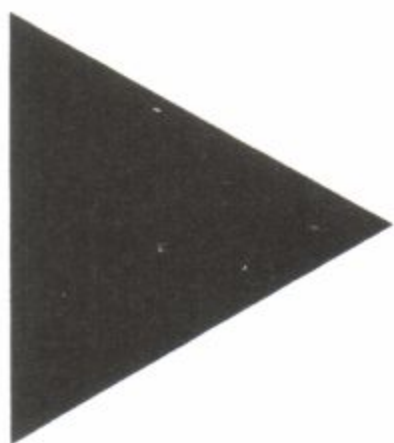
| Bdif.x | 内　容 | デフォルト |
|---|---|---|
| -I<ファイル名> | 差分ファイルに登録される入力ファイル名を <ファイル名> にする<br>(例) Bdif old.x new.x -Ioldold.x | |

| | | |
|---|---|---|
| -O<ファイル名> | 差分ファイルに登録される出力ファイル名を ＜ファイル名＞ にする<br>(例) Bdif old.x new.x -Onewnew.x | |
| -A | 差分データを追加する | |
| -D | ドキュメント等の差分をとる | |
| -W | 比較サイズの設定<br>16バイト単位で指定する | -W96 |
| -M | 比較マッチレートの設定<br>単位は % | -M75 |
| -L | 調査範囲の設定 | -L20 |
| -P | 連続変更箇所の最大値の設定<br>単位は Kバイト | -P16 |
| -S | 比較バイト単位を1バイトにする | 2バイト |
| -E | 差分抽出に必要なメモリ量の表示 | |
| -R | 逆差分情報の付加 | |
| -B | 差分情報を冗長にする | |
| -F | 作業経過の逐次表示 | |

| Bup.x | 内　　容 |
|---|---|
| -R | リバースパッチを行う |
| -L | 差分ファイルの記録情報を表示する |
| -LM | 差分ファイルの記録情報を表示する注1 |
| -E | 拡張子 .LZH のファイル(差分ファイルを含むもの)に対して、差分ファイル以外のものを展開して書き出す |
| -C | 元ファイルのチェックを行う |
| -I | 元のファイルのディレクトリを指定する |
| -O | 書き出すディレクトリを指定する |
| -NT | タイムスタンプをチェックしない |
| -NC | CRC をチェックしない注2 |
| -NL | ファイル長をチェックしない注2 |
| -NA | アトリビュートをチェックしない |

注：
1) ファイル名は12文字までに制限される。
2) なるべく-NC、-NLは使わないようにしてください。

デフォルトでの仕様を説明すると、2つのファイルを96バイト(比較サイズ、オプション“-W”)ごとに見比べていって75%(マッチレート、オプション“-M”)以上が同じものであれば、同一箇所だと判断して置換・挿入・削除をしていき、連続して16Kバイト(オプション“-P”)以上変更されていない限り、作業を続けるようになっています。

オプションに関しては標準が一番問題のない値となっていますので、むやみに指定しないほうがよいでしょう。筆者の場合も、オプションをつけたことはほとんどありません。

# Fu.x

## 便利で使いやすく、自由度の高いファイルユーティリティ

【概要】

ビジュアルシェルや、SX-WINDOWをお使いの方はともかく、コマンドシェルをお使いの方は、一度はコマンド入力が面倒だと感じられたことがありませんか。また、ファイル数が増えるにつれ、ディレクトリが深くなって移動が面倒になったり、ディレクトリ内のファイルの見通しが悪くなって、目的のファイルを探すことが大変だと感じている方もおられるかもしれません。

特にたくさんのファイルの中から特定のものだけを選んでコピーするなどということは、コマンドシェル上ではかなり面倒なことでした。Fuは、そういったファイル操作の面倒を軽減してくれるユーティリティです。ご存じの方もおられるかもしれませんが、FuはNECのPC-9801上で製作されたFD.EXEを参考にして作られています。このため、見た目はFD.EXEによく似ています。しかし、作者のかずりんさんがユーザの要望をさまざまな形で取り入れてくれたので、大変便利なユーティリティに仕上がりました。

たとえば、今、あなたがファイルをコピーしたいと考えたとします。従来のコマンドシェル上では、ファイル名がばらばらでワイルドカードが使えない場合、1つ1つのファイル名をCOPYコマンドに与えてコピーしていく必要がありました。しかし、Fuではあらかじめキーボード上の任意のキーにコピー機能を割り当てておけば、あとはコピーしたいファイルを選び、コピー機能を割り当てたキーを押してコピー先のディレクトリ名を指定すれば、一括して処理してくれます。

また、標準のコマンドにはない「ムーブ」（コピーしたあと、コピー元のファイルを削除してくれる機能）や、ファイルのソート、アトリビュートの変更など、従来のコマンドシェル上で行っていた操作のほとんどをFu上で行うことができます。

Fuの基本的な機能としては、

(1) ファイル操作に関する内部コマンドを持っている
(2) ユーザが任意のキーに外部コマンドを割り当てることで、簡単に外部コマンドを実行することができる
(3) 拡張子を自動的に判断し、ユーザが定義したとおりに拡張子に応じたコマンドを実行してくれる
(4) メニュー機能を持ち、定義した一覧の中から簡単にコマンドを実行できる
(5) 外部コマンド実行時にファイル名やオプションの受け渡し、画面の初期化などの面倒をFuが見てくれる

(6) Fu起動時の画面の色やファイルの表示形態などを変えることができる
(7) ユーザが簡単にFuの環境設定ファイルを書くことができる
(8) さまざまなFu用の支援ユーティリティ（エディタやビューワなど）が作られており、これらを外部コマンドとして、Fu上から起動できる柔軟さを持っている

などの機能を持っています。
　自分なりに環境設定をすませれば、ファイルユーティリティとしてだけではなく、コマンドシェル上でコマンド入力をする必要のほとんどない環境ユーティリティとしても使用することができるでしょう。

【作者】……………………☆かずりん☆　PEKIN　DEL
MAX BBS　0186
ねっとテロリスト　0079

【作者からの言葉】……　ども！、作者の☆かずりん☆です。
　私自身がコンピュータ初心者なので、環境ファイルなどの設定方法はなるべくわかりやすく作ってあるつもりです。だから、環境ファイルをバンバン書き換えて、正真正銘の自分専用のFu.xを作ってくださいね。パソコン通信をやっていない人は、PDSなどを目的で始めてみるというのでもいいですから、始めてみてください。結構おもしろくて病みつきになりますよ。それに、つねに最新のFu.xもパソコン通信で手に入りますし…。

【書式】……………………FU　［オプションスイッチ］
ECV　［オプションスイッチ］　環境ファイル名　環境設定する先のFu本体

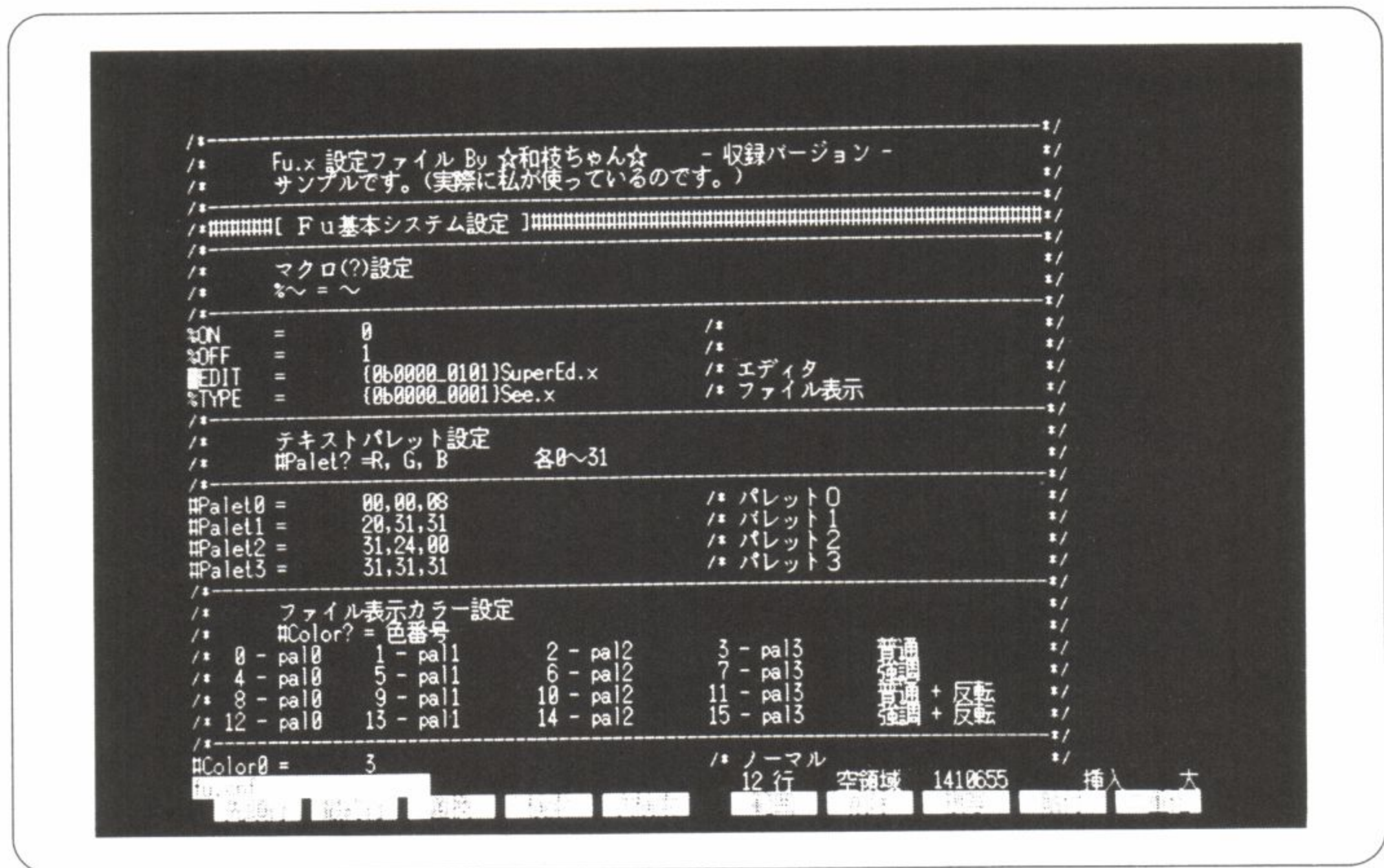
【インストール】………　まずは、解凍したFu関係のファイル（はじめて使用される方は、“Fu.x”、“Ecv.x”、“Fu.cnf”のみで結構です）を、あなたのディスクにインストールしましょう。インストール先は、あなたの好きなドライブ、ディレクトリでかまいません。
　フロッピーディスクで使用される方は、システムディスク上がよいでしょうし、ハードディスクユーザの方は、ユーティリティ関係の集まったディレクトリがよいでしょう。“Fu.x”と“Fu.cnf”は、同じディレクトリにインストールするということを忘れないでください。Ecv.xは、パスが通っていれば、必ずしも同じディレクトリでなくてもかまいません。インストールが終わったら、COMMAND.Xにパスが通っているかどうかを確認してください。

注：大半の方は大丈夫だと思いますが、筆者は以前これでハマったことがあります。Fuは、外部コマンドを使う場合、COMMAND.Xを起動します。通常はCOMMAND.Xにパスが通っていることが当た

り前になっていたため、何の意識もせずにFuを使用していました。ところが、ある日、Fuを使って環境を変更したところ、Fuから外部コマンドが全然立ち上がらなくなってしまったのです。理由がわからず、筆者は非常に慌てたのですが、実はファイルを移動した際、COMMAND.Xをパスの通っていないディレクトリに移動させていたのでした。

パスが通っていなかったら、AUTOEXEC.BATをエディタで編集します。“PATH=……”と書かれた行の先頭にでもCOMMAND.Xのパス名を書き加えてください。先頭に近ければ近いほど検索動作が速くなり、起動時間も短縮されますので、COMMAND.Xのような重要なものはできるだけ前のほうに書いておきましょう。

さて、解凍されたばかりのFuは環境設定がされていません。いっしょにコピーした“Fu.cnf”がFuに与える設定の書かれたファイルです。まずは、これをエディタで編集してみましょう。

**Pht.1** Fu.cnfをED.Xで読み込んだ画面

付属の“Fu.cnf”は、作者の使用されている環境だそうで、なにやら、さまざまな数値と命令が並んでいます。ざっとスクロールさせていくと、ファイルの後半には見たことのあるようなコマンドや、聞いたこともないコマンドがあるかもしれません。とりあえず、ここでは見るだけにして、ファイルの先頭に戻ってください。

11行目に“%EDIT”、12行目に“%TYPE”という文字列があります。{}に囲まれた、よくわからない数字が並んでいますが、その後ろに“SuperED.x”と“SEE.X”があります。これは、11行目がFuから起動する「エディタ」、12行目がファイル(特にドキュメントなどですが)を読むための「ビューワ」の設定になっています。ここをふだんあなたが使用している「エディタ」や「ビューワ」に書き換えてみてください。

たとえば、今まで標準のED.Xを使用されていたのならば、“SuperED.x”を“ED.X”に

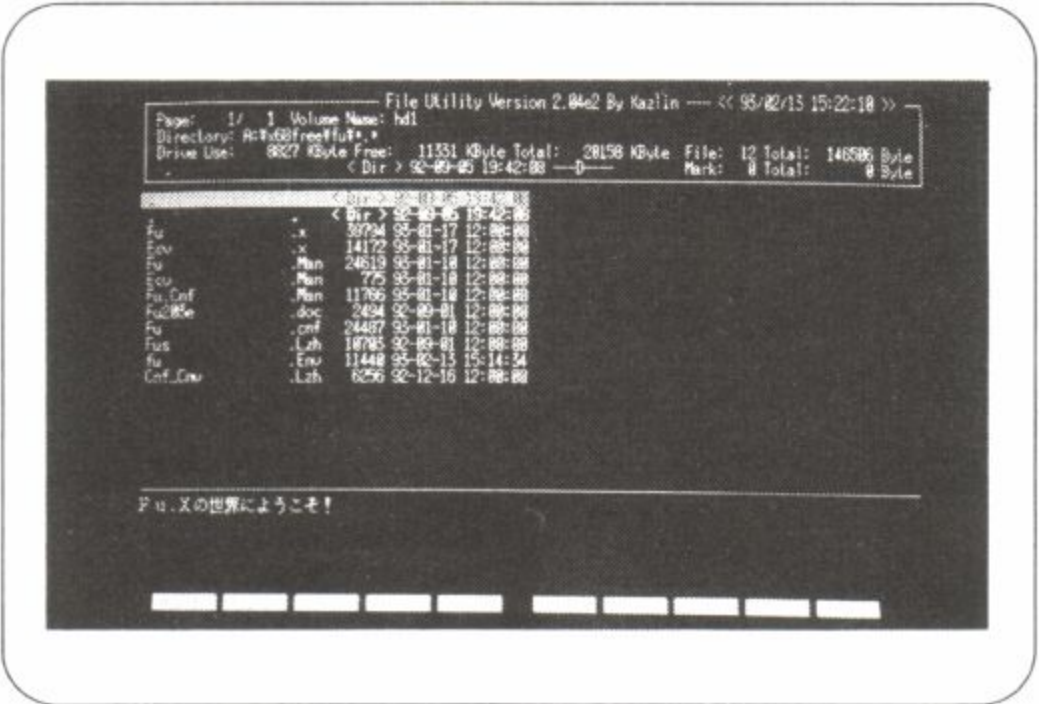
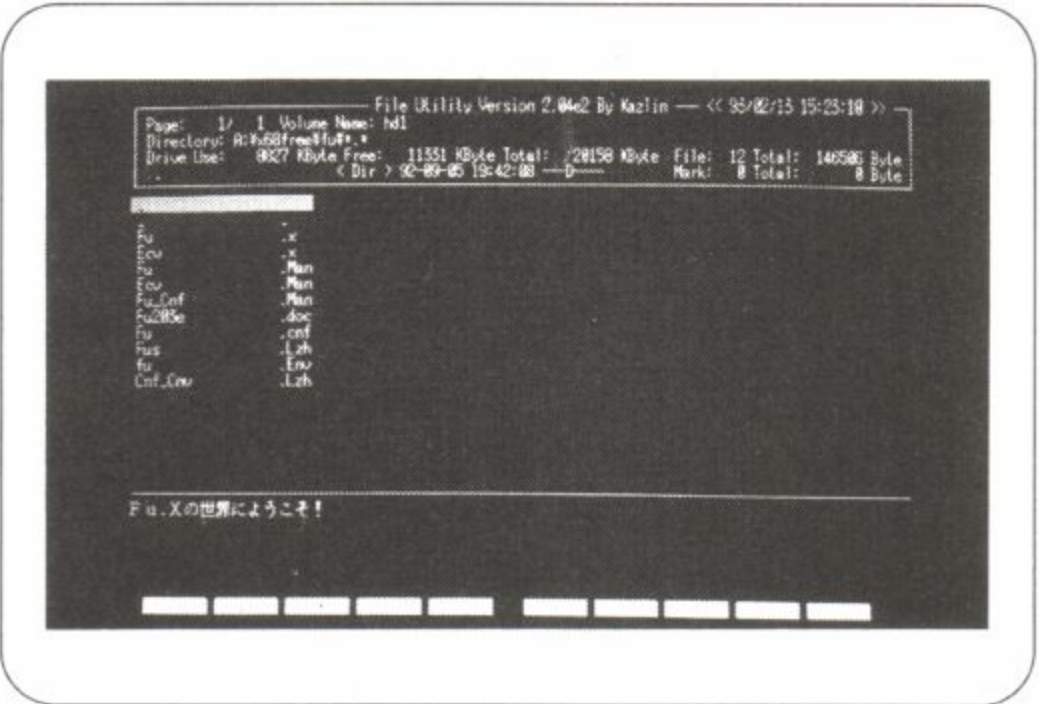
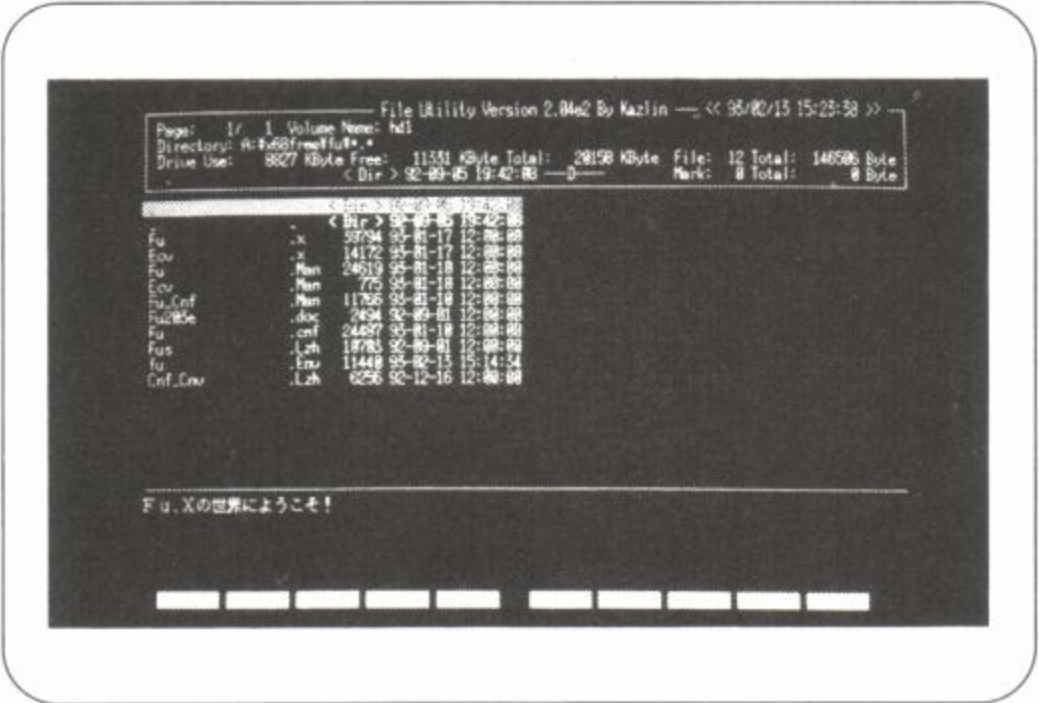
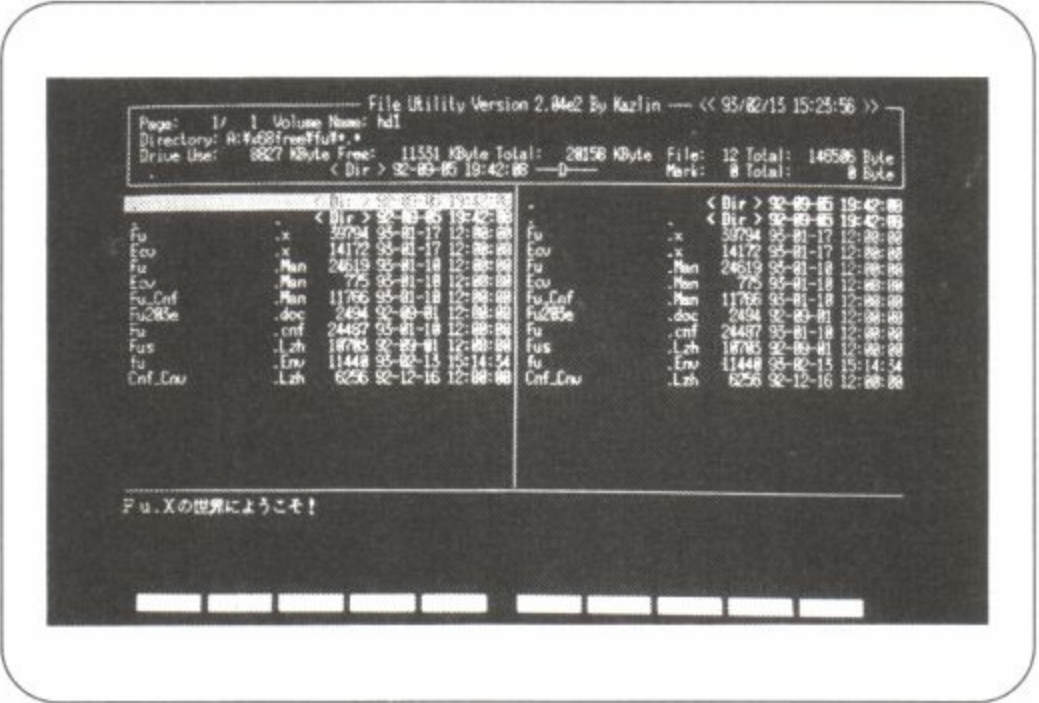
書き換えてもいいでしょう。ビューワとしてフリーソフトの“view.x”を使用されているのならば、“SEE.X”を“view.x”に書き換えてもいいでしょう。もちろん、“SuperED.x”、“SEE.X”はどちらも今回のディスクに収録されているので、この機会に両方ともあなたのディスクに入れてしまえば、わざわざ設定を変更する必要はないかもしれません（このFu.cnfは、今回の収録ディスク用に作成されたものです）。

なお、先ほども触れたように、これらの指定したファイルにはすべてパスが通っていなければなりません。パスが通っていない場合はフルパスで指定してください。フルパスで指定すると、起動が速くなります。そのかわり、あなたの環境が変わるたびに、Fuの環境ファイルも書き換えなければならなくなるので一長一短はありますが。筆者は、フルパス指定をしないほうがのちのち便利ではないかと思っています。ですから、これらFuの環境ファイルに書くコマンドは、すべてパスが通るように注意してください。そうでないと、環境ファイルに書いてあっても、そのプログラムが起動しません。

以上の書き換えがすんだら、編集したファイルをセーブし、エディタを終了してください。そして、Fu関係のファイルがあるディレクトリに移動し、そこで、“Fu.x -S”と“-S”オプションをつけて実行してください。あとでまとめて述べますが、これはFuの環境ファイルをテキスト形式からFuの本体が解釈できるバイナリ形式にコンバートするためのものです(ちなみに、付属のコンバートコマンド、Ecv.xを呼び出しています)。他に何もコンフィグファイル“Fu.cnf”をいじっていなければ、正常に終了するはずです。そして、Fuのあるディレクトリに“Fu.env”というファイルができていることを確認してください。もし作られていないときは、ディスクの容量不足か、他にコンフィグファイルをいじった可能性があります。確認してみてください。

**【オプションスイッチ】**　それでは、Fuを使用するときのオプションスイッチについて説明します。

### ●Fuのオプションスイッチ

| オプションスイッチ | 内　容 |
|---|---|
| -S | 環境ファイルをFuのバイナリファイルに変換する<br>例）　FU -S［環境ファイル名］ |
| -Xn | Fu動作時のファイル表示の列数を設定をする<br>n には2または4の値を指定<br>2の場合、2列で表示され、ファイルの情報（タイムスタンプ、ファイル容量など）が細かく表示され、4の場合、4列でより多くのファイルを見ることができるが、ファイル名のみの表示になる<br>例）　FU -X2 |

**Pht.2**　-X2の場合の起動画面　　-X4の場合の起動画面

| | |
|---|---|
| -Wn | Fu動作時のウィンドウ分割モードを指定する<br>n には０～２の値を指定<br>０の場合、標準の１画面に処理中のディレクトリを表示し、１、２の場合、２画面に分割して、処理中のディレクトリとコピーやムーブの移動先になるディレクトリを表示し、ファイル処理の確認をしながら作業できる<br>１と２の違いは、右と左のどちらが初期状態で処理をしているディレクトリになるかを設定するもの（１＝左、２＝右）<br>例）　FU -W1 |

**Pht.3**　-W0の場合の起動画面　　-W1の場合の起動画面

| | |
|---|---|
| -Dn | n には０または１を指定<br>Fuは通常X68000が使用していない画面モードをCRTCを操作することによって使用している。このため、純正のディスプレイの中でも表示が乱れたりするなどの不都合が出る場合がある。このスイッチは、この機能をキャンセルし、通常の画面モードでFuを使用するためのもの。通常は０だが、１にすることで、通常の画面モードを使用する<br>例）　FU -D1 |

Fuは、その内部機能として、オプションで指定するものと同等のものを持っているので、環境ファイル中でこれらの内部機能を設定しておけば、起動後でも任意に変更できますし、環境ファイル中で指定しておけば、毎回Fuの起動時に面倒なオプションスイッチをつけることもなく、気に入った画面で使用することができます。

●Ecv.xのオプションスイッチ

| オプションスイッチ | 内　　容 |
|---|---|
| -S | Fuの環境ファイルをバイナリファイルに変換する<br>なお、Ecv.xは標準で環境ファイルを変換する機能を持つため、オプションをつけなくても機能する<br>例)　ECV -S FU.CNF FU.X<br>　　　ECV FU.CNF FU.X　(上と同じ働きをする) |
| -V | 環境ファイルを表示しながら、環境ファイルを変換する |

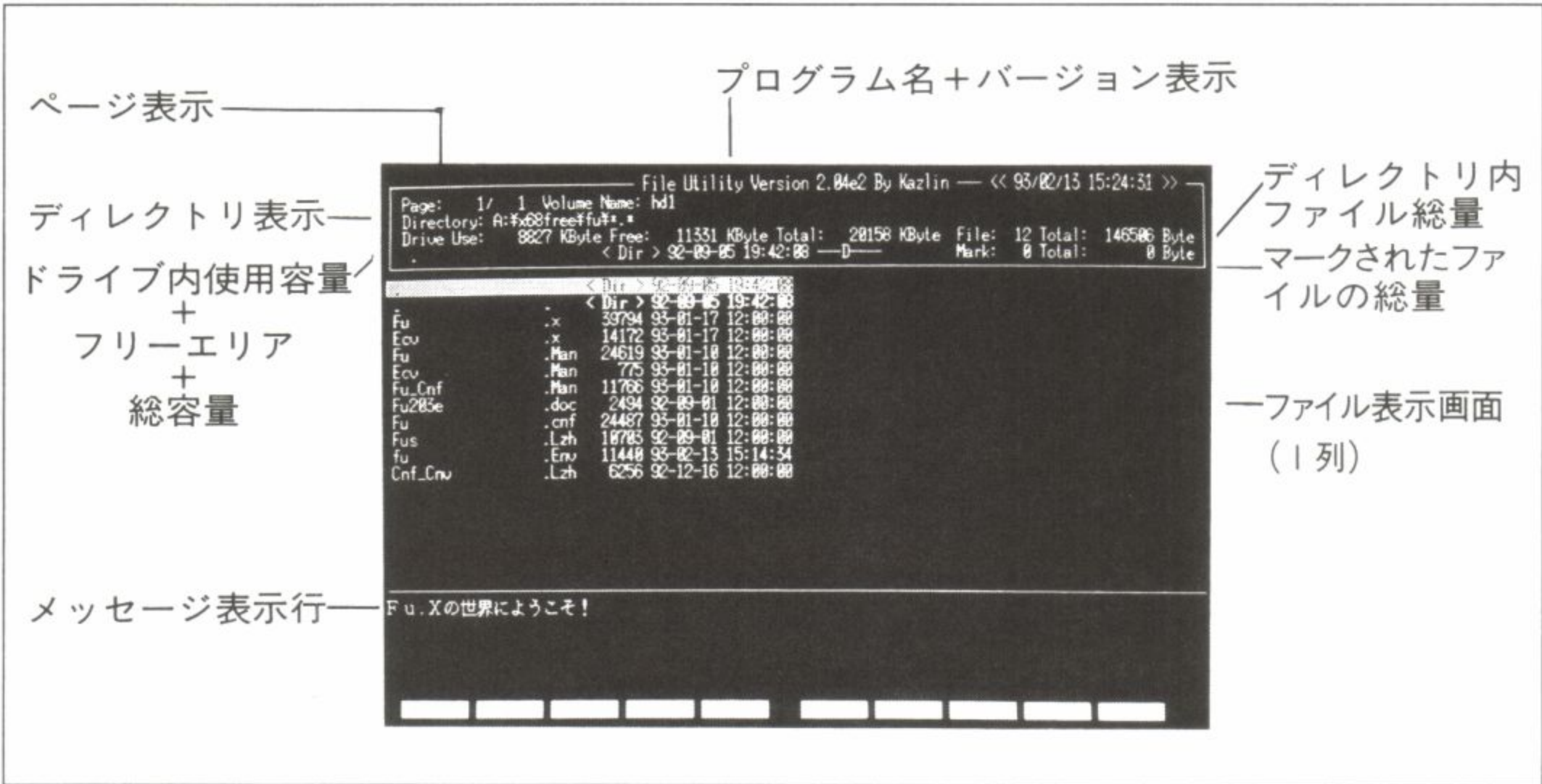
【基本操作】…………

## 1. 起動

それでは、Fuを起動してみましょう。

最初は“FU[CR]”と、何もオプションを指定せずに起動しても結構ですし、先に説明したように、好みによってオプションを指定してもかまいません。

Pht.4　Fuの起動画面

ページ表示
プログラム名+バージョン表示
ディレクトリ表示
ドライブ内使用容量
+
フリーエリア
+
総容量
ディレクトリ内ファイル総量
マークされたファイルの総量
ファイル表示画面(1列)
メッセージ表示行
Fu.Xの世界にようこそ!

無事に起動できたでしょうか？　もし表示が乱れたり、文字が欠けたりする場合は、ディスプレイが拡張画面モードに対応していないと思われます。いったんFuを終了し、“-D1”オプションをつけて起動し直してください。終了する場合は、[ESC]キーを押して、そのまま[CR]キーを押してください。

とりあえず、現在のFuの設定はディスクに収録されていたサンプルのものです。今後、あなたが設定ファイルを書き換えることで機能が変更されるかもしれないことを了解しておいてください。

## 2. カーソルの移動

Fuのメイン画面であるファイル表示画面には、ファイルを反転表示するバーがありま

す。Fu起動時にはファイルの先頭にこのバーが表示されているはずです。これをFuでは、「カーソル」と呼びます。ファイル操作時にファイルを指定するために使用されます。このカーソルは、その名のとおり、標準状態では「カーソルキー」によって上下左右に操作可能です。また、テンキー上の数字キーでも、ゲームのように操作することができます。通常ファイル画面の端まで来てもまだファイルが存在する場合、画面が切り替わって残りのファイルが表示されます。

下のディレクトリに移動したい場合は、カーソルがどのファイルの位置にあってもかまわず[CR]キーか[SPACE]キーを押してください。逆に上のディレクトリに行きたいときは[BS]キーを押してください。また、[¥]キーを押せばルートディレクトリまで一気に戻れます。

### 3．ドライブを変更する

Fuを起動した直後の画面は、Fuを起動させたディレクトリを表示しているはずです(Pht.4参照)。Fu.xやFu.cnfなどのファイル名も見えますね。それでは、今度はドライブを変更してみましょう。ドライブの変更は、[CTRL]キーとドライブ名のキーを同時に押せば変えることができます（Pht.5参照)。

**Pht.5** ドライブB:の表示

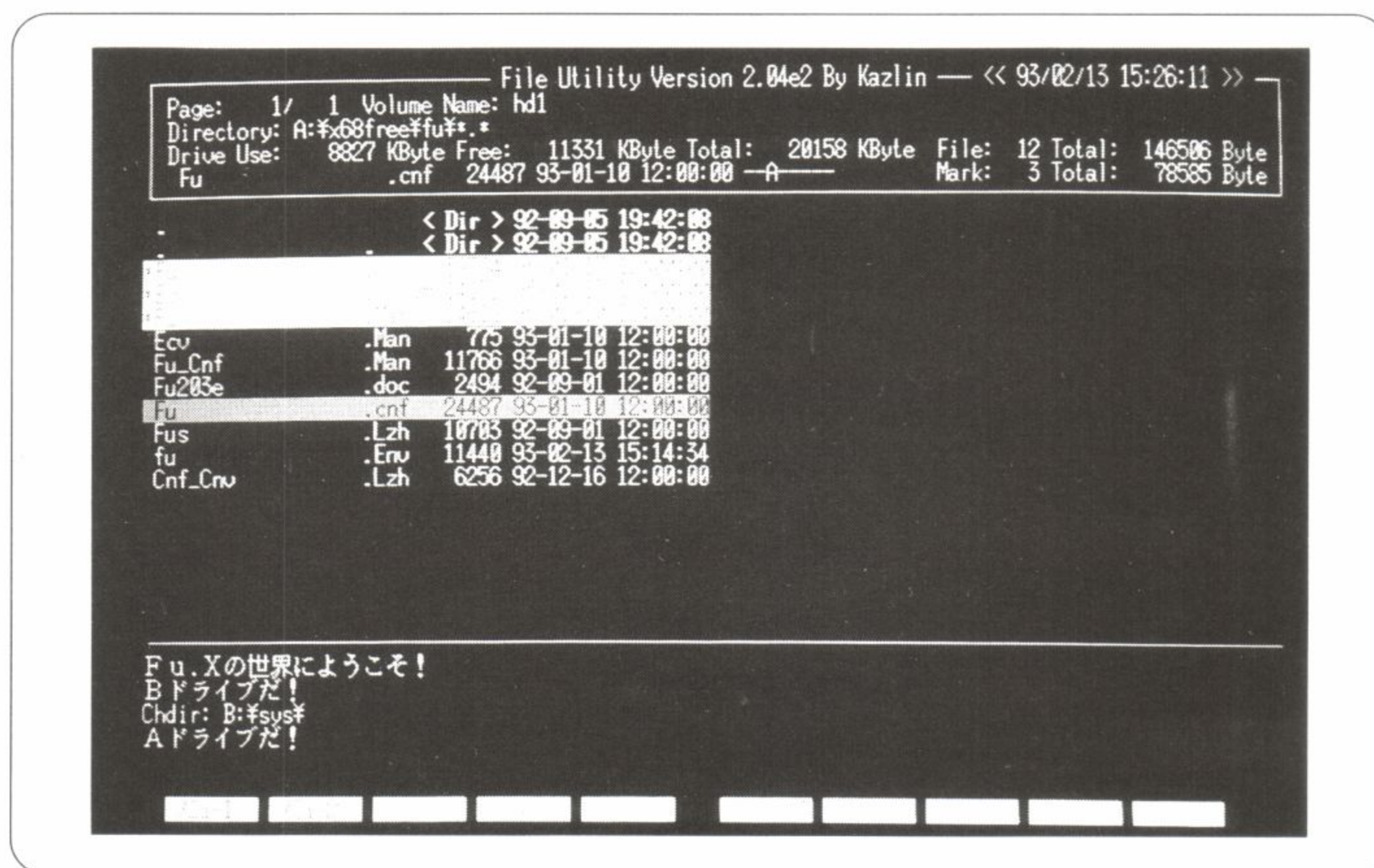
```
File Utility Version 2.04e2 By Kazlin — << 93/02/13 15:25:37 >>
Page:    1/   1  Volume Name: hd1
Directory: A:¥x68free¥fu¥*.*
Drive Use:    8827 KByte Free:    11331 KByte Total:    20158 KByte  File:   12 Total:  146506 Byte
 Cnf_Cnv           .Lzh    6256 92-12-16 12:00:00 —A———          Mark:    0 Total:       0 Byte

.                    < Dir > 92-09-05 19:42:08
..                   < Dir > 92-09-05 19:42:08
Fu           .x        39794 93-01-17 12:00:00
Ecv          .x        14172 93-01-17 12:00:00
Fu           .Man      24619 93-01-10 12:00:00
Ecv          .Man        775 93-01-10 12:00:00
Fu_Cnf       .Man      11766 93-01-10 12:00:00
Fu203e       .doc       2494 92-09-01 12:00:00
Fu           .cnf      24487 93-01-10 12:00:00
Fus          .Lzh      10703 92-09-01 12:00:00
fu           .Env      11440 93-02-13 15:14:34
Cnf_Cnv
     A:hd1                          (SASI HDD)
     B:hd2                          (SASI HDD)
     D:No Volume Name               (   2HD   )

Fu.Xの世界にようこそ!
Bドライブだ!
Chdir: B:¥sys¥
Aドライブだ!
```

### 4．ファイルをコピーする

Fuはファイルユーティリティですから、ファイル操作機能は充実しています。それでは実際にファイルをコピーしてみましょう。

まず、コピーしたいファイルにカーソルを移動させます。そのファイルにカーソルを置いたら、[C]キーを押してください。「＊（ファイル名）＊をコピーします」「コピー先は」と表示されたウィンドウが開いたはずです。ここでコピー先のディレクトリパス名をキーボードから入力してください。入力が終わったら、[CR]キーを押します。これでファイルがコピーされました。うまくコピーされたかどうか、コピー先のディレクトリに移動して確認してみてください。ムーブも同様の操作で実行できます（[M]キーを押す)。

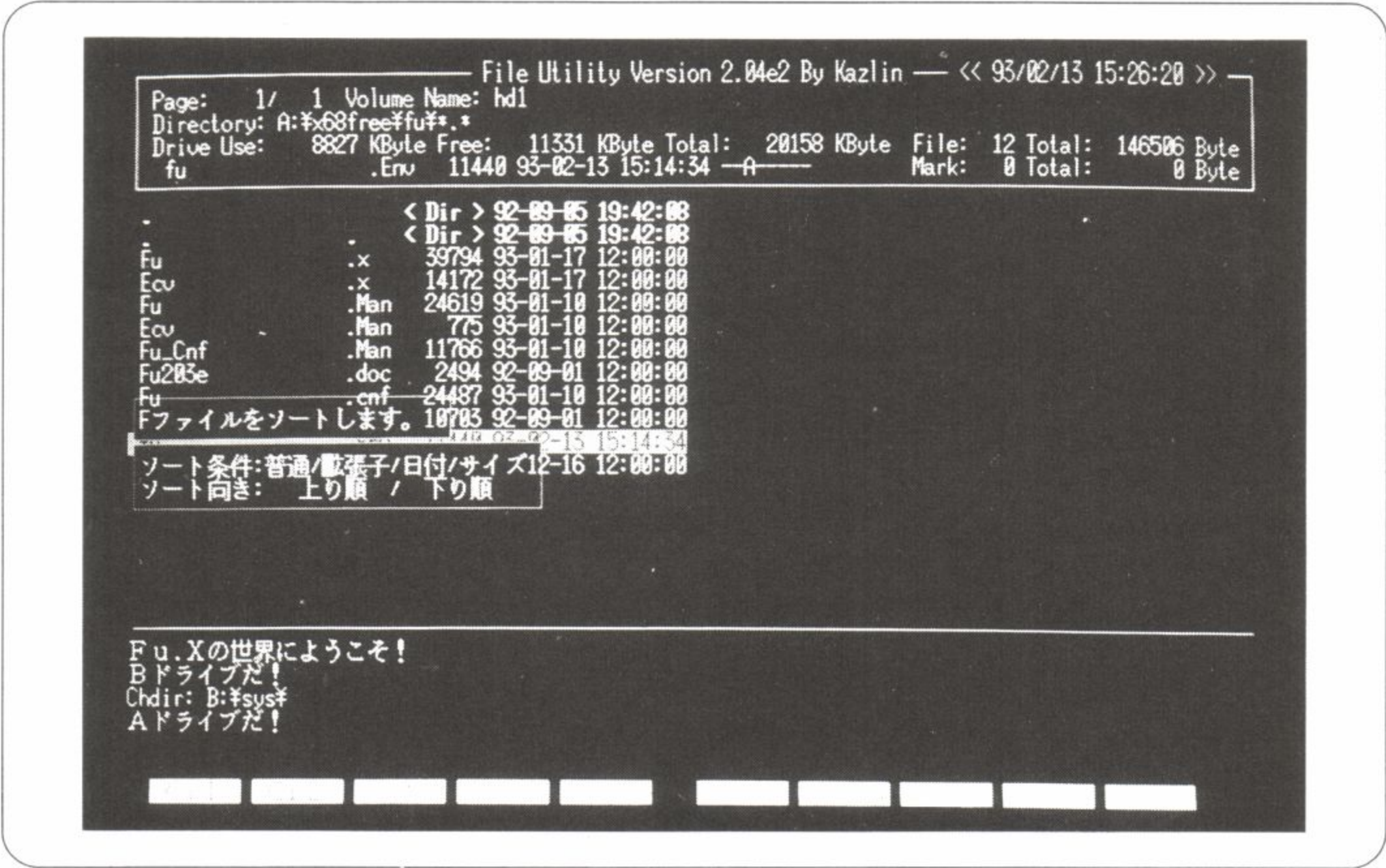
また、Fuではファイルを1つずつコピー（ムーブ）するだけではなく、複数のファイルをまとめてコピー（ムーブ）することができます。その場合、カーソルは1つしかありませんから、ファイル名の上にカーソルを置くのではなく、マークといってファイルに印をつけていきます。目的のファイルにカーソルをあわせたら、[SPACE]キーを押してください。ファイルが反転表示されましたね（Pht.6)。

Pht.6　マークされたファイル

マークするファイルはいくつでもかまいません。マークしたあとは、すべてのファイルが通常のコピーと同様の操作でまとめてコピー（ムーブ）されます[注]。また、カレントディレクトリにあるファイルをすべてコピー（ムーブ）したい場合は[HOME]キーを押してください。すべてのファイルがマークされます。マークしたファイルを解除したい場合は、もう一度[HOME]キーを押せば解除されます。

注：Fuでは、マークされたファイルのコピーや移動をマークしていった順番に実行します。

## 5. ファイルをソートする

カレントディレクトリ内のファイルをソートさせる場合は、[S]キーを押してください。小さなウィンドウが開きます（Pht.7)。

Pht.7 ソートウィンドウ表示

メニュー内のカーソルはカーソルキーで動かせますので、目的のソート条件に設定して[CR]キーを押してください。うまくできましたか？

以上、簡単ですが、Fuの基本操作を紹介してみました。この調子でFuの操作方法を書いていくと紙数が何枚あってもきりがないので、これくらいにしておきます。詳しくは、Fuのドキュメントを見てください。

**【カスタマイズ】**……… それでは、Fuの環境をあなたの使いやすいように設定していきましょう。ファイラというのは、自分の使いやすいようにカスタマイズすることでより使いやすくなっていくものですから。ただし、いきなり新規に環境ファイルを書くよりは、Fuに付属してきた環境ファイルを書き換えたほうが楽でしょう。

それでは、カーソルキーを“Fu.cnf”にあわせてください。ファイル数が多すぎて画面にすべてが表示されていない場合でも、カーソルを下か右に移動させていけば、表示しきれなかったファイルも表示されます。カーソルをあわせたら、そこで[E]キーを押してみてください（Pht.1参照)。

最初に設定したように、あなたの設定したエディタが“Fu.cnf”を読み込んで起動したは

ずです。すぐにFuの画面に戻ってしまった方は、エディタにパスが通っていなかったり、Fuのコンフィグファイル、Fu.cnfに指定したエディタのファイル名が間違っていないかどうかを確認してみてください（本書に添付されているFu.cnfはエディタとしてSuper-EDが指定されています。SuperEDはフォントファイルがないと「フォントファイルを読み込めません」というメッセージを出して起動できません。フォントファイルがないままSuperEDを起動させるには、[OPT.1]キーを押しながら起動させてください。詳細はSuperEDのページを参照）。

無事にエディタが起動した方は、勝手知ったるエディタ上でFu.cnfを眺めて見てください。Fuの環境ファイルはわかりやすく、自由度の高い点が特徴だと思います。

それでは、このサンプル版を参考にしてFuの機能を説明していきますので、あなたなりの環境設定をしてみましょう。

## 1. 環境設定

大きく分けてFuで設定できる機能は、「システム関係」「内部コマンド」「外部コマンド」の3つに分けられます。サンプルの環境ファイルもこの順番で並んでいますので、ここでも、この順番に話を進めます。

なお、Fuの環境ファイルは、すべて「機能（またはFuに設定するキー）＝命令」という形になっており、大変わかりやすくなっています。

### ●システム関係の設定

サンプルの環境ファイルの先頭から書かれている「システム関係の設定」には、Fuの環境を設定するコマンドが並んでいます。なお、サンプルの設定ファイルと、この文章中では、Fuの機能がつかみやすいように大文字、小文字で書き分けてありますが、実際のFuは大文字、小文字で処理を分けることはありませんので、厳密に設定ファイルの大文字、小文字を書き分ける必要はありません。

#### マクロの設定

環境ファイル全体の中で繰り返し設定される機能を、簡単な命令に置き換えて設定の手間を省くためにあります。これを、作者は「マクロの設定」と呼んでいます。Fuでは任意のキーに機能を割り当てられるようになっていますが、「[E]キーにエディタを割り振って、[F5]にも割り当てよう」と考えたとき、いちいちエディタの名前や、オプションなどを繰り返し書いていたのでは面倒です。しかも、使用するエディタを変えたとき、環境ファイル中のすべてのファイル名を変更していかなければならず、手間ばかりかかります。

しかし、“%EDIT”のように設定しておきさえすれば、[E]のキーでも[F5]のキーで

も、“%EDIT”と書くだけでエディタが立ち上がりますし、プログラムを変更したときも、この設定の行を書き換えるだけでプログラムの変更は終わってしまいます。

あなた自身で好きなように“%～”と設定してみてください。とりあえず、“%EDIT”(エディタの設定)、“%TYPE”(ビューワの設定)とだけ設定しておけば十分でしょう。

### テキストパレットの設定

Fuが起動したときの画面の色を設定します。使用できるパレットは4色です。「ファイル表示カラーの設定」と組み合わせて、Fuの画面イメージを決定します。あなたの好きな色に設定してください。Fu上部のタイトル、時計は、ここで決められたカラーで表示されます。テキストパレットの設定のしかたは、「Human68kユーザーズマニュアル」に解説されていますので参照してください。

### ファイル表示カラーの設定

上記のパレットと、ここで設定する属性(強調、反転など)とを組み合わせ、ファイルの表示カラーを設定します。

| カラー番号 | 内　容 |
|---|---|
| #Color0 | 通常のファイル注1 |
| #Color1 | リードオンリー属性のファイル |
| #Color2 | 不可視属性のファイル |
| #Color3 | システム属性のファイル |
| #Color4 | ディレクトリ |
| #Color5 | ボリュームネーム |
| #Color6 | リンクファイル注2 |
| #ColorE | カーソル注3 |
| #ColorF | マーク |

注：
1）#Color0～5については、「Human68kユーザーズマニュアル」を参照のこと。
2）これは、フリーソフトの“lndrv.x”によって、論理的にそのディレクトリに結びつけられているファイル。
3）Fuでファイルを指定するためにユーザが動かすバーを「カーソル」、指定したあと、画面につけられる印を「マーク」という。

以下の部分は、Fu内部での処理の設定をする部分です。なお、初期値となっているものは、この行を“Fu.cnf”などの環境ファイルに書かなかった場合、Fuが自分で設定する値です。

| 機能 | 内　容 | デフォルト |
|---|---|---|
| /* | コメント<br>行の先頭にこれをつけることで、その機能をコメント化し、使用しないことができる。内部機能のときは、初期値が使用される | |

| | | |
|---|---|---|
| #??? | 拡張子実行の区切記号<br>Fuは、ファイルを指定されたとき、その拡張子に応じて外部コマンドを起動する機能を持っている。このとき、他にキーを押しておくことで複数（最大で4つ）のコマンドを起動することができる。設定のときは、拡張子に続いてコマンドを4つ並べられるが、このときのコマンドごとの区切りに使用するキャラクタを指定するもの。<br>特別不都合がない限り、設定する必要はないと思われる | ## |
| #Disp_Mod | 特殊な画面モードを使用するかしないかを設定する命令。初期値は0（使用する）で、100桁×32行の画面を使用している。これによって1画面中の情報量は増えるが、メーカが公開した画面モードではないので、ディスプレイによっては表示が乱れることがある。画面が乱れてしまう場合は、この設定を1にすれば、従来の画面モード(96桁×32行)を使用するようになる。この機能は、オプションスイッチ“-Dn”と同等の働きをする | 0（使用する） |
| #Fu_Alias | Fuの持つ簡易エイリアス機能を使用するかしないかを設定する<br>初期値は0(使用する)。使用しない場合は1に設定する[注1] | 0（使用する） |
| #File_Mod | ファイル表示の形態を設定する<br>オプションスイッチ“-Xn”と同等の働きをする。指定できる値は2または4で、2のときは画面に2列にファイル名が並ぶ。また、ファイル容量、タイムスタンプも表示される。4のときは4列にファイル名が表示される。たくさんのファイルを表示する分、ファイル容量、タイムスタンプは表示されない。たくさんのファイルがあるディレクトリで目的のファイルを探すときは4に、通常の操作時は2でよいだろう | 2 |
| #Key_Rt0<br>#Key_Rt | Fu使用時のキーリピートの開始時間とリピート時間を設定する<br>値が少ないほど、速く反応する。自分のマシンの処理速度を好みにあわせて快適になるように設定したほうがよい | Rt0=20、Rt1=2 |
| #Time_Xpos<br>#Time_Ypos<br>#Time_Color | 時計表示の座標指定などの設定。それぞれ、時計表示の「X座標」「Y座標」「表示色」を設定する。初期値は、画面の右上の枠上に表示<br>「表示色」は、「ファイル表示カラー設定」に従う | Xpos=72<br>Ypos=0<br>Color=3 |
| #Time_Switch | 時計表示をするかしないかを設定する<br>表示したくない場合は1 | 0（表示する） |
| #Copy_Input | コピーまたはムーブの機能で、転送先のディレクトリパス名を入力するウィンドウの大きさを設定する[注2] | 32 |
| #Box_Color | Fuでは、命令の実行前に確認をとったり、ユーザに入力を求めたりすることがある。そのとき、画面に入力用のウィンドウが開かれるが、そのウィンドウの枠の色を設定する。色は「ファイル表示カラーの設定」のパレットに従う | #Color1 |
| #Input_Max | Fuは、定義された命令実行時にユーザにオプションなどの入力を求めることがある。これは、その入力を求めるときに開かれるウィンドウの大きさを設定する | 32 |
| #Over_Copy | コピーやムーブのとき、転送先に同一の名前を持つファイルがあった場合、Fuはどうしたらよいか、ユーザに確認をとってくる。このときのFuの確認ウィンドウの初期値を設定する<br>数値は0から5までで<br>0……そのままコピー、ムーブ<br>1……タイムスタンプを比較し、新しければ実行する<br>2……コピー、ムーブをしない<br>3……コピー、ムーブするはずのすべての指定ファイルを実行する<br>4……コピー、ムーブするはずのすべての指定ファイルのタイムスタンプを比較して新しければ実行する<br>5……ファイル名を変更して実行する | 1（タイムスタンプが新しければ、コピーまたはムーブする） |

| | | |
|---|---|---|
| | 0から2までと5は、複数のファイルがコピー、ムーブの指定になっていても、転送先に同じファイル名があれば、そのつど確認をとってくる。3と4は1回指定すれば、すべてのファイルを指定された処理で実行する注3 | |
| #Exec_Chk | 実行コマンドの確認ウィンドウの初期値を設定する<br>初期値に設定されたものは、ユーザが不用意に実行されては困る外部コマンドの場合、実行確認のためのウィンドウを開くことができる。これによって、不用意にコマンドが動作することを防止できる。数値は0から3まで設定可能<br>0……そのまま実行する<br>1……実行しない<br>2……すべて実行する<br>3……終了する | 1（実行する） |
| #Del_Chk | 削除コマンドを実行した場合、誤ってファイルを消去しないように、もう1度ユーザに確認をとる。その確認に与える動作の初期値を設定する。初期値に指定された動作は[CR]キーのみで実行される。設定可能な数値は0（削除する）、1（削除しない） | 1（削除しない） |
| #Zcopy_Chk | コピーやムーブのとき、ユーザが転送先として指定したディレクトリパス名が存在しないとき、Fuは、そのドライブに新たに転送先として指定されたディレクトリを作成してよいかどうかを確認してくる。その確認に与える初期値を設定する。設定可能な数値は0（ディレクトリを作成する）、1（ディレクトリを作成しない） | 1（ディレクトリを作成しない） |
| #File_Push | この機能が有効になっていると、Fuは現在処理をしているディレクトリから別のディレクトリに切り替えられる際に現在処理しているディレクトリ中のカーソルの位置を記憶し、再びそのディレクトリにユーザが戻ったとき、カーソルを移動前の場所に表示する。この機能が無効になっていた場合、元のディレクトリに戻ったとき、カーソルはファイル群のトップに表示される<br>0ならば有効になり、この機能が働く。1ならば無効 | 1（無効） |
| #Drive_Color | 内蔵機能"/Drive"を使用すると、ドライブ変更のメニューウィンドウが開く。このウィンドウの中で表示されるカーソルの色を指定する。色はファイル表示カラーの設定に従う | |
| #Drive_Switch | 内蔵機能"/Drive"コマンドを使用したとき、ドライブ変更のメニューウィンドウ内のカーソルの動きを設定する。好みによって変更すること。初期値は0。1の場合は、[→][←]のキーを押しても上下と同じく1ステップずつカーソルが移動する | 0（カーソルキーの[→][←]でメニューの最大・最小にカーソルが移動する） |
| #File_Win | 画面モードの設定をする。Fuのオプションスイッチ"-Wn"と同等の処理をする。0の場合、Fuは1画面にファイルを表示する。1、2の場合、画面を2分割し、対向画面を参照しながら作業を行うことができる。このとき、コピーなどのファイル操作も、指定先は対向画面が初期値となる。1と2の指定の違いは、画面の左右どちらが初期のアクティブ画面となるかの違い。詳しくは"-Wn"を参照 | 0（1画面にファイルを表示する） |

| | | |
|---|---|---|
| #Win_Pos | Fuでは、ファイル操作や内部機能でウィンドウを開いて、Fuからユーザへ入力を求めたり、確認を求めたりする。また、外部機能もユーザの設定次第で同様の確認を求めさせることが可能。このとき、開くウィンドウの位置を設定する<br>0の場合、カーソルがある位置付近にウィンドウを開く。ユーザは、Fu上でファイル操作をする場合、カーソルの動きを目で追っているので、そのすぐ近くに確認のウィンドウが開き、視点の移動が少なくてすむ。1の場合、カーソルの位置にかかわらず、画面の中央にウィンドウが開かれる。つねに決まった位置にウィンドウを開くため、ユーザはウィンドウを見逃しにくく、また、あらかじめウィンドウの開く位置を予想できるという利点がある | 0（カーソル位置にウィンドウを開く） |
| #Write_Chk | 内蔵機能“/File_Write”コマンドを使用したときの実行確認の初期値を設定する注4<br>0の場合、そのまま[CR]キーを押せば、ディスクにファイルの並びが記録される。1（書き込まない）にしておけば、ユーザが[Y]キーを押さない限り、（誤って違うキーを押しても）ディスクには書き込まれない | 0（ディスクに書き込む） |
| #Copy_Shift | コピー、ムーブ時の転送先のディスク容量の残量の表示の設定をする<br>初期値は0（表示する）で、通常は残量がFuの上部のタイトル部にリアルタイムで表示される。また[SHIFT]キーを押すことで、残量の表示、計算を省略し、コピーやムーブの動作を高速にすることができる。1（表示しない）を指定すると、通常はディスクの残量を計算しないでコピーやムーブの作業を行う。容量が足りるかどうか心配な場合は、[SHIFT]キーを押せば、残量を計算して表示してくれる。ハードディスクなどの大容量のメディアで作業を行う場合は残量の心配をあまりすることはないので、この設定は1でもいいかもしれない | 0（表示する） |
| #Version | Fuのタイトル部分のバージョン表示を変更する。通常は作者のバージョンナンバーが表示されている。内容に変更を加えた場合以外は変えないほうがよい | |
| #End_Chk | Fuを終了する際に、終了の確認をさせることができる。これによって、誤ってFuを終了しないようにすることができる。この確認メッセージに与える初期値を設定する。初期値は0（終了する）。0に設定した場合、確認メッセージが表示されたとき、[CR]キーを押すだけで終了することができる。逆に1（終了しない）に設定したときは、[Y]キーを押さない限り終了しない | 0（終了する） |
| #Exec_Mod | Fuの機能の中に「拡張子を判断し、ユーザが設定したとおりに拡張子に応じたコマンドを実行してくれる」機能があることは最初に紹介した。Fuの特徴的な機能で、大変便利なものだが、この機能が働く拡張子のファイルがカーソル位置にあるファイルの拡張子を判定するか、マークされたファイルの拡張子を判定するかを設定する。<br>初期値は0で、カーソル位置のファイルの拡張子を判定する。1にすることでマークされたファイルの拡張子を判定する | 0（カーソル位置のファイルの拡張子を判定する） |
| #Pop_Mod | Fuは同一ディレクトリ内のファイルをユーザが自由に並べ替えることができる機能（/File_Pop）を持っている。このとき、指定したファイルを1度ファイル群から抜き出して、再び任意の位置に挿入するという形で並べ替えを行う。この挿入のときに「カーソルのある位置に挿入」するか、「カーソルの下の位置に挿入」するかを設定する<br>初期値は0（カーソルの位置に挿入する）。1のときは「カーソルの下の位置に挿入」となる注5 | 0（カーソルの位置に挿入する） |

| | | |
|---|---|---|
| #Menu_Max | Fuは、ユーザが自由に外部実行コマンドや内部コマンドを登録して実行することが可能だが、これらのコマンドを単にキーボード上のキーに1つずつ割り当てるだけではなく、メニュー形式にしてまとめて登録し、そのメニューウィンドウから選択することもできるようになっている。#Menu_Maxは、このメニューウィンドウの中に表示できる文字列の長さを設定する | 32文字（半角） |
| #Mes_Sw | Fuは、実用性とはあまり関係はないが、一定時間キー入力がないと、メッセージを表示することができる。こういう実用性以外の楽しさがあるところがFuが広く使われる理由なのかもしれない。初期値は0で、メッセージが表示される。1に設定しておくと、メッセージは表示されない。メッセージの表示がうるさい人は、1にしておくとよい。しかし、Fuを楽しく使いたい人は、「表示する」に設定しておいたほうがいいだろう | 0（メッセージが表示される） |
| #Msc? | 上記のメッセージの表示間隔の時間設定。“?”は、時間設定（秒）と#Mes?で設定するメッセージの番号で、0～3まで設定可能<br>#Msc0=0<br>#Msc1=190<br>#Msc2=230<br>#Msc3=254<br>#Msc0に設定された時間に#Mes 0が表示され、#Msc1に設定された時間に#Mes 1が表示される。以下同様に#Msc2、#Msc3と設定された時間に、#Mes 2、#Mes 3が表示される。時間が0ならば、起動時にメッセージが表示される。255ならば、そのメッセージは表示されない。それ以外の値の場合は、キー入力がなくなってから、設定した時間がたつごとにそれぞれのメッセージを表示する | |
| #Mes? | メッセージになる文字列を設定する。“?”は0～3までで、それぞれ表示間隔を設定する#Mscの番号に対応するメッセージとなる<br>#Mes0=(Fu.Xの世界にようこそ！)<br>#Mes1=(あ、なんか無視されてません？)<br>#Mes2=(そう・・・嫌いなのね・・・)<br>#Mes3=(ぐ、ぐれてやるぅ！)<br>また、メッセージではなく、実行コマンドを設定しておくことで、設定された時間がたつとコマンドを実行させることもできる。たとえば、“[COPY BEEP.SYS PCM”と設定しておけば、メッセージのかわりにBEEP音を鳴らす。スクリーンセーバという画面の焼き付きを防ぐプログラムや、画面のコントラストを下げるプログラムを用意しておけば、実用的な用途にも使える | |
| #Fu_Sw | Fuの動作時にファンクションキーを表示しておくかどうかを設定する。初期値は0（表示する）で、1だと表示しない | 0（表示する） |
| #Fn? | ファンクションキーに表示する文字列を設定する。作者添付の設定ファイル、Fu.cnfを参照すること。基本的にはファンクションキーに割り当てた機能の名前を書いておくという使い方が一般的だろう。もちろん、ユーザが自由に設定できるので、自由に設定してかまわない | |
| #Fu_Dot | 通常Fuのファイル画面の先頭に表示されているディレクトリマーク“.”にカーソルがあり、そこで[CR]キーが押されたときの動作を設定する。初期値は0で、ルートディレクトリに移動する。1に設定すると、ディレクトリの移動は行われず、ファイルの再検索を行う | 0（ルートディレクトリに移動する） |
| #Fu_Color? | Fuがユーザに入力を求めるときに開くウィンドウの各部分の色を設定する。“?”は0～2で、0はウィンドウの枠の色、1はFuからのメッセージの表示色、2はユーザに求める値の初期値の表示色になる | |

注：

1）ディレクトリの深い環境を構築した方は、Fuのようにコピーやムーブのときに、ディレクトリ

パス名の入力を求める方法は面倒だと感じられるかもしれない。Fuは、ユーザの入力の手間を軽減するために内部にエイリアス機能を持っている。これは、たとえば、“&DATA=B:¥DATA”と設定しておくだけで、コピーのとき、ディレクトリパス名を求められたら“DATA”と入力するだけで、“B:¥DATA”にコピーされる機能である。便利な反面、“DATA”とつくものは、すべて“B:¥DATA”に展開されてしまうため、設定するときは他のファイル名などと重複しないように注意すること。

2）ユーザは、ディレクトリパス名を入力し、Fuに指定するが、その入力のためのウィンドウの大きさを設定する。初期値は32。通常はこれだけあれば十分だが、深いディレクトリ環境を構築していたり、入力文字数の設定が半角のため、全角でディレクトリ環境を構築していて全角16文字で足りないという方は、この値を大きくすること。逆にウィンドウが大きくて、ファイル表示が隠れて邪魔だという方は数値を小さくすること。

3）なお、あくまでも確認のためのウィンドウの初期値の指定なので、確認のウィンドウが開いているときはすべての設定が利用可能である。初期値で設定されていれば、[CR]キーだけで実行できるため、よく使う機能を割り当てておけば便利である。

4）内蔵機能“/File_Write”は、ディレクトリやファイルの並びをディスクに書き込む機能。Fuでは、ファイルの並び順をユーザが自由に設定できるし、ファイルを自由な条件でソートして、表示、作業することが可能。しかし、これらのファイルの並び順は、Fuが並べ替えただけのもので、ディスク内にある実際のファイルの並び順は以前のままである。このため、1度ディスクになんらかの処理をしてFuが再びディスクのデータを読み込むと、ファイルの並び順は元に戻ってしまう。内蔵機能“/File_Write”は、ディスクにFuで並べ替えたファイルの並び順を書き込むことでユーザが理解しやすいようにファイルを並べ替える機能である。これによって、コマンドモード上でDIRなどのコマンドを使用しても表示されるファイルの並び順はFuで並べ替えたとおりのものになる。

ただし、“/File_Write”で書き込まれるファイルの並び順は、ディスクのディレクトリエントリという、いわばディスクの目次部分を書き換えるだけなので、ディスク内にあるファイル本体の並べ替えまでは行ってくれない。なお、この機能は、ディスクに書き込むという、一歩間違えれば危険な機能を実行するため、確認をとるようになっている。

5）言葉だと説明しにくいものなので、設定を変えてみて、どういう動作をするか試してみること。どちらでも、あなたの感覚にあうほうを選べばよいだろう。

## 2. 内部コマンドの設定

さて、これでFuの基本的な環境の設定は終わりました。これらの環境は、実用性とともに、Fuを使用するときに、快適かつ使いやすいように用意されているものです。Fuの特徴として、これらのユーザの設定できる環境が多い点も挙げられるでしょう。

では、Fuを便利に使いこなすための設定をしていきましょう。Fuは、大変自由度の高い機能の割り当てが可能なため、最初のうちはとまどうかもしれませんが、自由度が高いだけにユーザの好みにあった環境を構築することが可能です。

定義自体の書式は大変簡単ですし、定義のしかたを間違えている場合は、設定ファイルの登録時（“Fu -S”や“Ecv.x”を使用時）にエラーメッセージを表示して、定義を間違えている行番号などを教えてくれます。いろいろ定義を変えて試してみてください。

内部コマンドの定義のための書式は、基本的に次のようになっています。

**コマンドを割り当てるキー　＝　Fuの内部コマンド**

「コマンドを割り当てるキー」には、当然ながら、キーボード上のキーを使用します。基

本的にキートップに表示されている文字を使用します。また、[SHIFT]キーは"SFT"、[CTRL]キーは"CTL"、[OPT.1][OPT.2]キーは、それぞれ"OPT1"、"OPT2"と表記し、他のキーと組み合わせてコマンドを割り当てることができます。これらのキーと他のキーとを組み合わせるときは、"+"を使って表します。

また、[ROLL UP]、[ROLL DOWN]キーは、それぞれ"R_UP"、"R_DOWN"となり、[CR]キーは"RET"、[SPACE]キーは"SP"、[ろ]キーは"_"、フルキーとテンキー部の両方にあるキーは、テンキー部分を指定するときは( )でくくります。たとえば、テンキー部の[/]キーを使用したいときは"(/)"と表記します。言葉で説明すると難しいように感じられるかもしれませんが、添付の設定ファイルを参照していただければ、すぐに理解できるでしょう。

例)

a = /File_Attrib　(Fuの内部コマンド、ファイルのアトリビュート変更機能を[A]のキーに割り当てる)

SFT＋q = /Fu_End　(Fuの内部コマンド、Fuの終了機能を、[SHIFT]キーと[Q]キーが同時に押されたときに働くように設定する)

SFT＋CTL＋d = /Drive　(Fuの内部コマンド、ドライブ変更機能を、[SHIFT]キーと[CTRL]キーと[D]キーが同時に押されたときに働くように設定する)

コマンドを割り当てるキーの指定方法は、基本的には以上のような感じです。サンプルの設定ファイルを見ると、[ESC]キーなど、以上の説明にあわない書式がありますが、それについてはあとで述べます。

**●Fuの内部コマンド一覧**

| 内部コマンド名 | 内　容 |
|---|---|
| /Csr_Up | カーソルを1つ上に移動する |
| /Csr_Down | カーソルを1つ下に移動する |
| /Csr_Right | カーソルを1つ右に移動する |
| /Csr_Left | カーソルを1つ左に移動する |
| /Csr_Min | カーソルをアクティブディレクトリの一番先頭のファイルに移動する |
| /Csr_Max | カーソルをアクティブディレクトリの一番最後のファイルに移動する |
| /Mark_Set | カーソルファイルをマークする |
| /Mark_Set2 | カーソルファイルをマークする。ただし、ディレクトリはマークしない |
| /Mark_All | アクティブなディレクトリのすべてのファイルをマークする |
| /Mark_All_r | アクティブなディレクトリのすべてのファイルを反転マークする(マークされていたものはマークをはずし、マークされていないものはマークする) |

| | |
|---|---|
| /Fu_End | Fuの終了。Fu起動時のカレントディレクトリに戻す |
| /Fu_End0 | Fuの終了。Fu起動時のカレントディレクトリに戻さない |
| /Fu_End_Chk | 終了確認付きの"/Fu_End" |
| /Fu_End0_Chk | 終了確認付きの"/Fu_End0" |
| /Dir_Up | 現在のディレクトリから1つ上の階層のディレクトリに移動する |
| /Chdir | 現在のドライブのカレントディレクトリに移動する |
| /Dir_Move_exe0 | カーソルがディレクトリにあれば、その下層へ移動する。ファイルならば、ユーザの割り当てた拡張子実行外部コマンドに処理を移す |
| /Dir_Move_exe1 | カーソルがディレクトリにあれば、その下層へ移動する。ファイルならば、ユーザの割り当てた拡張子実行外部コマンドに処理を移す |
| /Dir_Move_exe2 | カーソルがディレクトリにあれば、その下層へ移動する。ファイルならば、ユーザの割り当てた拡張子実行外部コマンドに処理を移す |
| /Dir_Move_exe3 | カーソルがディレクトリにあれば、その下層へ移動する。ファイルならば、ユーザの割り当てた拡張子実行外部コマンドに処理を移す注1 |
| /Dir_Move_Mark | カーソルがディレクトリにあれば、その下層へ移動する。ファイルならば、マークする |
| /Mark_Push | 現在のマークを記憶する |
| /Mark_Pop | 記憶したマークを復帰する |
| /File_Copy | マークされたファイル、なければカーソルファイルをコピーする |
| /File_Move | マークされたファイル、なければカーソルファイルを移動する注2 |
| /File_Delete | マークされたファイル、なければカーソルファイルを削除する注3 |
| /File_Attrib | マークされたファイル、なければカーソルファイルのファイルアトリビュートを変更する注4 |
| /File_Time | マークされたファイル、なければカーソルファイルのタイムスタンプを変更する注5 |
| /File_Rename | マークされたファイル、なければカーソルファイルのファイル名を変更する。キーボードから変更するファイル名を入力する |
| /File_Mask | ファイルの検索条件を指定する。すると、該当以外のファイル名を持つファイルは表示されなくなる。検索には、ワイルドカードを使用する。ワイルドカードについては「Human68k Ver.2.0　ユーザーズマニュアル」を参照のこと。再び、「全てを検索」に戻す場合は、ワイルドカードに"*.*"を指定する |
| /File_Search | ファイルを検索する。検索条件にはワイルドカードを使用できる。検索か、検索したファイルをマークするかをたずねてくるので、指定する。検索の場合は、該当するファイルを1つ1つカーソルで指示して確認してくる。マークをする場合は、該当するファイルを1つ1つカーソルで指し示して確認し、マークする |
| /File_Sort | ファイルを指定の条件にそって並べ替える注6 |
| /File_Mod | Fuのファイル表示形態の変更注7 |
| /File_Win0 | ファイル表示ウィンドウの表示変更<br>1画面とするか、2画面分割にするかを変更する |
| /File_Win1 | 分割したウィンドウ間でカーソルを移動する |
| /File_Win2 | Fuの表画面と裏画面を入れ替える。表画面と裏画面は、それぞれ独立してファイルマーク情報などを保持する |
| /Drive | カレントドライブの変更をメニュー形式で行う。メニューはカーソルキーで選択する。[→][←]のカーソルキーの動作は、内部設定#Drive_Switchの値によって変化する |
| /Page_Up<br>/Page_Down | アクティブなディレクトリのファイルが1画面にすべて収まらない場合、1画面ごとに移動させる |
| /Gr_On0 | 512×512ドット　65536色のグラフィック画面を表示する |
| /Gr_On1 | 768×512ドット　16色のグラフィック画面を表示する |
| /Gr_On2 | 640×400ドット　16色のグラフィック画面を表示する |
| /Gr_On3 | 512×512ドット　256色のグラフィック画面を表示する |
| /Gr_On4 | 512×512ドット　16色のグラフィック画面を表示する |
| /Gr_On5 | 1024×848ドット　16色のグラフィック画面を表示する |
| /Gr_On6 | 800×512ドット　16色のグラフィック画面を表示する注8 |

| /Env_Read | この機能が働くと、FuはFu.envファイルを再読み込みし、Fuの設定を再定義する |
|---|---|

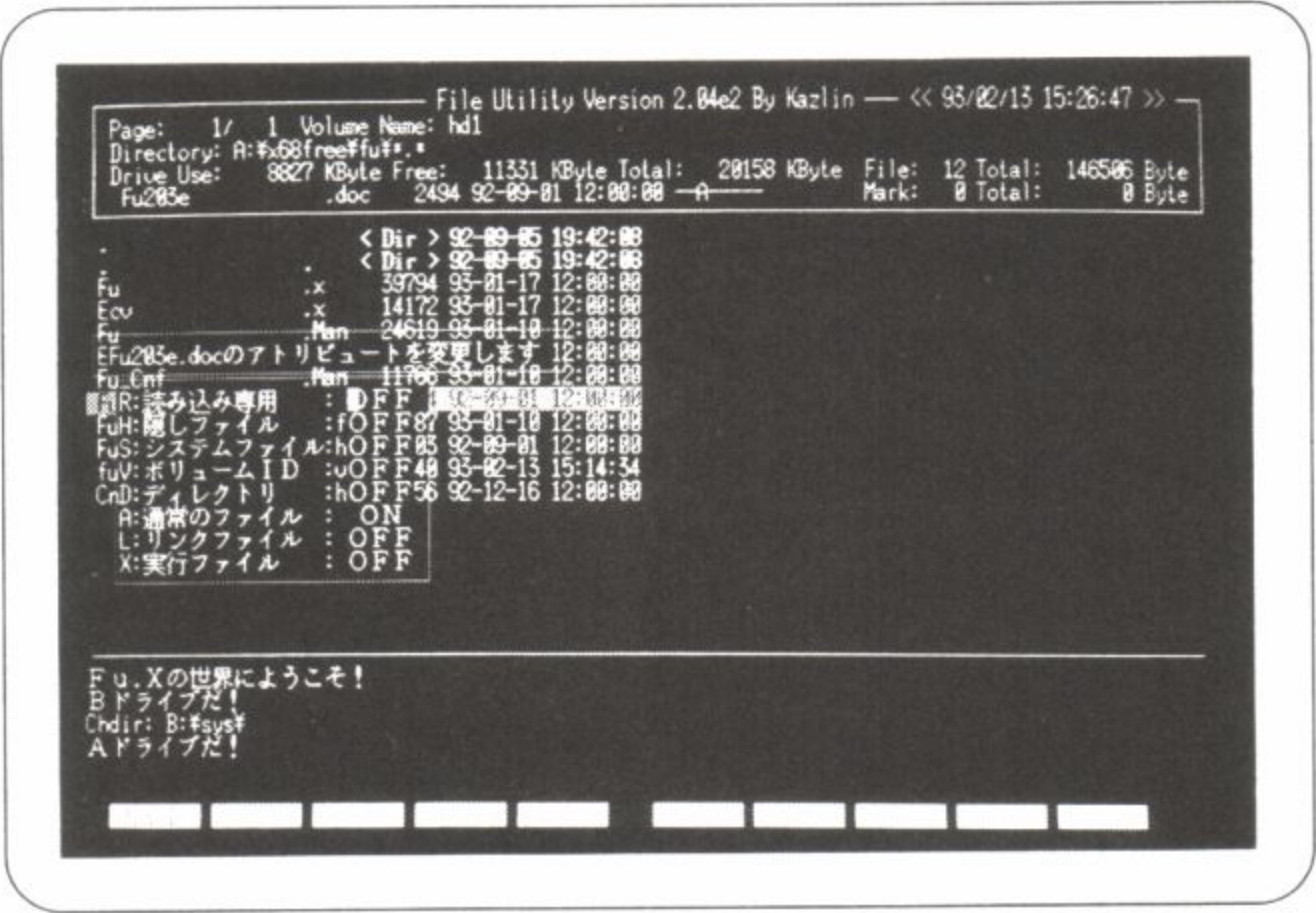
注：

1）以上、4つの処理の違いはない。ここで割り当てた4つの起動キーによって、最大4つの外部コマンドを同一の拡張子のファイルに対して使い分けることができるようになる。

2）コピー、ムーブ機能が働くと、転送先をたずねるウィンドウが開くので、キーボードから転送先のディレクトリ名を入力して[CR]キーで確定すること。

3）ファイル削除機能が働くと、削除してもよいかどうかの確認ウィンドウが開く。削除してもよいのなら[Y]、削除しないのならば[N]を入力すること。Fuの内部設定、#Del_Chkの値が初期値となる。

4）アトリビュート変更機能が働くと、現在のファイルのアトリビュート一覧が表示されたウィンドウが開くので、カーソルキーで各アトリビュートのオン／オフを切り替えて[CR]キーで確定すること（Pht.8）

**Pht.8** アトリビュート変更ウィンドウ

5）タイムスタンプ変更機能が働くと、現在のファイルのタイムスタンプが表示されたウィンドウが開かれるので、直接数字を入力して変更するか、カーソルキーで数値を増減させて[CR]キーで確定すること。

6）ソート機能が働くとソートの条件をたずねるウィンドウが開かれるので、カーソルで指示すること。ソートが終了すると、画面がソート順に並べ替えられる。

7）Fuは、最初に述べたように、オプションスイッチや、内部設定によって1画面に表示するファイル情報、ファイル数を変更できる。しかし、Fu起動後に表示形態を変更する場合は、この内部コマンドを設定しておく必要がある。

8）上記の設定のうち、24KHzモードのないディスプレイでは使用できないモードがある。また、シャープにより正式にサポートされていない画面モードを使用している場合もあるため、表示が乱れる場合がある。

## 3．外部コマンドの登録

Fuは上記の内部コマンドだけでもファイルメンテナンスにはかなりの力を発揮します。しかし、Fuを本当に便利で手放せないツールにしているのは、Fuから外部コマンドを起動できるという点にあります。外部コマンドの登録方法は、基本的には内部コマンドと同様です。つまり、

**登録した外部コマンドを起動させたいキー　=　外部コマンド**

例)

m = mxdrv.x

となります。もちろん、これら登録した外部コマンドは通常コマンドラインから起動しているコマンドであり、特別Fu用に改造したりする必要はありません。

また、外部コマンドの多くがオプションを必要としますが、よく使用するオプションはあらかじめ設定に入れておくなり、外部コマンド起動のたびにオプションを使用するかどうかをたずねるようにするといった使用状況に応じた動作をFuがサポートしてくれます。

なお、外部コマンドの起動方法には、このような「キーによって外部コマンドを起動する」方法以外に、「拡張子に応じたコマンドを起動する」といった方法も用意されています。たとえば、X68000のミュージックデータのうち、最もデータ量が豊富と思われる“MXDRV.X”（みるく氏ほか）対応のデータの場合、拡張子が“.MDX”で統一されています。ユーザは、“.MDX”の拡張子を見るだけで、「これは、MXDRV.X対応のデータだ」とわかります。しかし、ユーザが“.MDX”の拡張子を見て、その実行コマンドである“MXP.X”を起動し、そのファイルを読み込むというのでは今ひとつ不便です。できれば、その拡張子に応じた実行ファイルを起動し、そのファイルを読み込んでくれたら大変便利ですね。これが、「拡張子による外部コマンドの実行」です。これによって、ユーザはファイルを指定するだけで、目的のファイルを利用することができます。

Fuから外部コマンドを起動できるだけでも便利ですが、さらに外部コマンド起動時にFuがさまざまな機能を用意することで、さらに便利で快適に使用することができるようになります。これらの機能は、それぞれの機能を組み合わせることも可能です。これら外部コマンドの起動を工夫し、充実させることで、あなたのFuはますます快適で使いやすいものになっていくでしょう。ただし、環境整備に最も力を入れなければならないのも、この外部起動コマンドの部分です。

### ●Fuの外部コマンドサポート機能

それでは、外部コマンドをFuから起動するときのサポート機能から説明しましょう。

| 表記 | 機　能 |
| --- | --- |
| ^ | ディレクトリ変更<br>“^”に続いてドライブ名、ディレクトリ名を設定することで、そのドライブ、ディレクトリに直接移動することができる<br>例）CTL＋A = ^A:<br>　　CTL＋F1= ^C:¥SYS |

| | |
|---|---|
| ^^ | ディレクトリ変更<br>“^”と同一の機能。機能上異なる点は、“^”は同一ドライブでの移動の場合、マークしたファイル名をFuが記憶したままディレクトリを移動するので、移動先にマークしたファイルと同名のファイルがあると、その移動先の同名のファイルがマークされた状態になる。これに対し、”^^”の場合は、同一ドライブであろうと、移動前にマークしたファイルをクリアしてからディレクトリを移動するので、移動先に同名のファイルが存在してもマークはされない注1<br>例） SFT＋A = ^^A:<br>SFT＋F1= ^^C:¥SYS |
| [ | コマンドシェル上のECHO OFFと同一機能<br>Fuは、通常外部コマンドを起動する場合、画面下部のコマンド行に起動する外部コマンドのファイル名を表示するが、外部コマンドの先頭に“[”をつけることで、この表示をしないようにすることができる<br>例） I = [MXDRV.X -p500 |
| %p | 現在のパス名に展開される<br>外部コマンドに現在のパス名を与えるときに使用する |
| %f | 現在カーソルのあるファイル名に展開される<br>外部起動コマンドのあとに指定することで、ファイル名の指定などに使用できる。たとえば、COPYコマンドのあとに“%f”を指定しておけば、COPYコマンドが呼び出されたときに、カーソルのあったファイルがコピーされる。たとえば、カーソルが“IOCS.X”の上にあった場合、COPYコマンドが呼び出されると、“IOCS.X”がコマンドに渡される<br>例）E = SuperED.X %f<br>P = COPY %f PCM |
| %e | “%f”と同様、現在カーソルがあるファイル名が外部コマンドに展開されて渡される<br>“%f”が拡張子までファイル名として外部コマンドに渡されるのに対して、“%e”は拡張子を除いた部分のみ外部コマンドに渡される点が異なる。起動される外部コマンドが拡張子を省略したファイル名のみで起動できる場合、また、外部コマンドに拡張子ごとファイル名が与えられると困る場合などには“%e”を使用する<br>例）Z = LZX.X %e<br>G = PIC.R %e |
| %m | マークされたファイルがあればそのファイル名を、マークされたファイルがなければ現在カーソルのあるファイルのファイル名を展開する<br>“%f”、“%e”との違いは、マークされたファイル名を展開する点が異なる。“%m”を指定することで、外部コマンド起動時にマークされたファイル名が与えられる。また、複数のマークファイルがあった場合には、マークされたファイル名は1つずつ外部コマンドに渡され、すべてのマークファイルが外部コマンドに渡されるまで外部コマンドを起動しつづける<br>例） SFT＋Z = LZX.X %m |
| %c | 複数のファイルをまとめて扱える外部コマンドを使用するときなどに使う<br>マークされたファイルがあれば、マークされたファイルをすべて並べた状態で展開し、外部コマンドに渡す。マークされたファイルがなければ現在カーソルがあるファイル名に展開する。たとえば、LHA.xなどで複数のファイルをまとめて1つの圧縮ファイルにまとめる場合などに使用する<br>例） L = LHA a %i”圧縮ファイル名は？” %c |
| %i、%l、%j、%k | Fuでは、コピーするときに転送先をたずねたり、エディタを起動するときに編集するファイル名をたずねたりする。このように、Fuからコマンドを起動するときに文字入力の要求をさせたい場合に使用する<br>また、「””」「##」「()」「{}」でくくってメッセージや入力文字列の設定をすることができる。それぞれの設定による違いはドキュメントを参照のこと<br>例） SFT＋E = SuperED.X %i”編集するファイル名は？” |

| | |
|---|---|
| - | 外部コマンド実行後、カーソルを1つ前のファイルに移動させる指定<br>Fuは、基本的にマークされているか、カーソルが置かれているファイルに対してコマンドを実行する。これは、コマンド実行後、カーソルを1つ前のファイルに移す機能。たとえば、1つのディレクトリに“.MDX”ファイルばかりをまとめていたとする。拡張子を判断して実行する外部コマンドで、そのディレクトリにある“.MDX”ファイルを、後ろから順番に前へと演奏させたいときなどに使用する。重複指定も可能で、“--”とすれば、2つ前のファイルへ……などと指定することもできる<br>例）　F = -MXP.X %f |
| + | 外部コマンド実行後にカーソルを後ろに移動させる指定<br>“-”はカーソルを前に移動させる指定だったが、こちらは、カーソルを1つ後ろのファイルに移動させるもの。やはり、重複して指定することができ、“++”などとすることで指定の数だけカーソルを進めることができる |

注：以上の機能上の違いを使い分けること。たとえば、“^”を使用する場合、ある同一ドライブのディレクトリにファイルをまとめてコピーするときに、コピーしたいファイルをマークしてからコピー先のディレクトリに移動すれば、同名のファイルが移動先でもマークされた状態になる。もし、すべて削除したい場合は、すでにマークされているので、内部機能の「ファイル削除」を使えば、マークされているファイルは一度に削除できる。また、ファイル名の重複をチェックする場合にも使えるだろう。通常のドライブ、ディレクトリの移動だけならば、“^^”を使用すればよい。

外部コマンドのサポート機能については以上のとおりです。

## 4．オプションビット

外部コマンド起動前後に外部コマンドの環境を設定する機能もあります。作者は、これを「オプションビット」と呼んでいます。これらの設定をするには、起動する外部コマンドの前に{ }で囲んだ数値を指定することで行います。この数値には「10進数」「16進数」「2進数」の数値表現が可能です。設定項目の実行時にFu.xのほうで内部形式に変換、実行してくれます。現在、使用できる数値範囲は10進数表記で「0～255」、16進数表記で「0x00～0xFF」、2進数表記で「0b00000000～0b11111111」です。あなたのわかりやすい表記で設定してください。筆者は、機能を組み合わせたとき、一目でわかりやすいという理由で2進数表記をよく用います。

現在使用できるオプションビットとその機能を示しておきます。

| 表記 | | | 機能 |
|---|---|---|---|
| (16進数表記) | (2進数表記) | (10進数表記) | |
| {0x00} | {0b00000000} | {0} | 通常の外部コマンド起動。Fuからは単に外部コマンドを起動するだけ。それ以外はなにもしない。デフォルトの設定なので、通常設定する必要はない<br>例）{0x00}[Ecv.x FU.cnf FU.X |
| {0x01} | {0b00000001} | {1} | 外部コマンド実行前に画面を初期化する<br>例）{0x01}[TMN.X |

| {0x02} | {0b00000010} | {2} | 外部コマンド実行後、キー入力があるまでFuに復帰しない。通常、Fuは外部コマンドを実行後ただちにFuに復帰する。このオプション機能が指定されると、ユーザが何かキーを入力するまで、Fuに復帰せずに待つ。画面にグラフィックデータなどを表示したときや外部コマンドの実行結果を確認したいときに使用する<br>例）{0x02}[+PIC.R %f<br>（この場合、画面初期化も含めて"{0x03}[+PIC.R %f"としたほうがよい） |
|---|---|---|---|
| {0x04} | {0b00000100} | {4} | 外部コマンド実行後、Fuでファイルを再検索する<br>たとえば、外部コマンド実行後、新しいファイルが作られたり、既存のファイルが削除されるような場合に使用する<br>例）{0x04}[COPY %f %i"コピー先のディレクトリ名は？" |
| {0x08} | {0b00001000} | {8} | 外部コマンド実行前に本当に実行してよいかどうかを確認してくる。ファイルの削除コマンドなど、誤って実行されては困る外部コマンドを実行するときに使用することで誤入力を避けられる<br>例）{0x08}[DEL %f<br>（この場合も、ファイル再検索も含めて、"{0x0C}[DEL %f"のほうがよい） |
| {0x10} | {0b00010000} | {10} | 外部コマンド実行時にファンクションキーを表示しない<br>ファンクションキー表示をしたくないとき、外部コマンド実行前に設定する<br>例）{0x10}[BUP -l %e |
| {0x20} | {0b00100000} | {32} | 外部コマンド実行前に、マークされていたファイルのマークをクリアする<br>例）{0x20}[LZX.X -d %m |
| {0x40} | {0b01000000} | {64} | 外部コマンド起動時にファイル表示画面をコンソール画面として使用するプログラム、"Fu_con.X"の機能をFuに内蔵させたもの注1<br>例）{0x40}[LHA L %e |
| {0x80} | {0b10000000} | {128} | 外部コマンド実行後画面を初期化する |
| {0xFF} | {0b11111111} | {255} | すべてのオプション機能を使用する |

注：Fuは、通常、画面下部を外部コマンドのコンソール画面として使用している。ただし、文字行にして5行程度しか確保されていないため、大量の画面表示（エラー表示やヘルプ機能など）をする外部コマンドの場合、表示が画面外に流れてしまい、ユーザはメッセージを確認することができなかった。ところが、この難点を解決すべくEmu氏が公開した、"Fu_con.X"という支援プログラム（Fuには、「Fu支援プログラム」とでもいうべき、Fuで使用されることを前提とした外部コマンドが公開されている。後述）を使うと、Fuのメイン画面というべきファイル表示画面を、外部コマンド起動時にコンソール画面として使用することができる。これによってコンソール画面の表示行数が20行ほど増えた。この"Fu_con.X"の機能を、Fuの作者が内蔵させたものが、このオプション機能。

もちろん、それぞれのオプション機能は組み合わせて使用することができます。その場合は、それぞれの数値を足してください。

## 5．拡張子判断実行コマンド

前述したように、Fuは、ある１つのキーに起動する外部コマンドを定義できるだけではなく、指定されたファイルの拡張子を判断して、起動すべき外部コマンドを選択する機能があります。起動の設定ができる外部コマンドは４つまでです。たとえば、".LZH"の拡

張子を持つファイル（LH.X、またはLHA.xで圧縮された圧縮ファイル）で、「LHA.xによる解凍」「LHA.xによる圧縮ファイル一覧」「圧縮ファイルへの追加」などを設定することができます。

書式は、前述の内部設定の最初にあった「拡張子実行の区切り記号」で区切って４つの外部コマンドを表記します（デフォルトの記号は「##」です）。

**.拡張子　＝　外部コマンド1##外部コマンド2##外部コマンド3##外部コマンド4**

例）.LZH = {0x04}[LHA E %f ## {0x40}[LHA L %f ## {0x44} [LHA A %f %i ''追加するファイル名は？''

となります。

また、外部コマンドが１つではなく、いくつものコマンドを連続して実行する場合など、１行が長くなって見苦しい場合には、

```
.拡張子　=　{：
　　　　　　外部コマンド1
　　　　## 外部コマンド2
　　　　## 外部コマンド3
　　　　## 外部コマンド4
　　　　　：}
```

のように、{：　：}を使って、複数行にわたって表記することも可能です。

これは、拡張子判断の外部コマンド実行時だけではなく、内部機能や、キー割り当てによる外部コマンドの起動の際にも、同様の表記をすることで、{：　：}でくくった内容を１行として扱うことが可能になります。

## 6．ワンステップ進んだ設定ファイルを作る

Fuの基本的な設定ができるようになったら、さらに一歩進んだ設定をしてみましょう。

### ●内部機能の連続表記

Fuの通常の内部機能は、

**起動するキー　＝　内部機能**

で設定できましたが、さらに内部機能1つだけの設定ではなく、いくつかの内部機能を組み合わせることも可能です。ただし、残念ながら、内部機能と外部機能を組み合わせて使用することはできません。ファイルをソートしたあと、ディスクにファイルの並びを書き出すことや、ファイル表示画面を1ページ送ってからカーソルを2つ下のファイルへ移動させるなど、簡単な内部機能のみを使ったバッチ処理が可能になります。

**●内部機能、外部コマンドのメニュー化**

Fuは、「1つの起動キーに1つのコマンド」という設定が基本でしたが、メニューという形で複数の内部機能や外部コマンドをまとめておき、「1つの起動キーを押すだけで複数のコマンドの中から必要なコマンドを選んで起動できる」という機能を使用することもできます。定義できるコマンドは、内部機能、外部コマンドともに最大で16個までです。

定義方法は、通常の定義内容の先頭に、メニューであることを示す"!"を加えてください。次に、メニュー画面を開いたときに何が登録してあるかがわかるようなメッセージを「" "」でくくってください。その後ろに登録したい内部コマンドや外部コマンドを書いてください。メニューには最大16個までコマンドを登録することが可能です。

メニューにするのですから、当然複数のコマンドを登録することになります。このとき、拡張子判断による実行コマンドのところで述べたように、設定ファイルを見やすくするために{:　:}でくくることで複数行にわたって表記することも可能です。

また、できればメニューのタイトルも設定しておくとわかりやすいかもしれません。メニュータイトルの書式は、"#タイトル名"です。

**書式**
**定義したいキー　=! "メッセージ1"コマンド1　## "メッセージ2"コマンド2　## … ## "メッセージ16"コマンド16**

通常複数行で表記したほうがわかりやすいですから、

```
定義したいキー＝{:
!  "メッセージ1"コマンド1
## "メッセージ2"コマンド2
##          ⋮
## "メッセージ16"コマンド16
:}
```

と表記したほうがよいでしょう。

例）

```
Y  {:
!     ''#音源ドライバの登録''
      ''MXDRV.Xの登録''pcm8.x || mxdrv.x -p500
##    ''ZMUSIC.Xの登録''pcm8.x || zmusic.x -p600
##    ''RCD.Xの登録''rcd.x
:}
```

以上のように、Fuの持つ多彩な設定方法を用いれば、快適で便利な環境を実現することができます。

注：Fuは、キー入力の自由度を得るために、ほとんどのキーのセンスを監視下に置く。このため、Fuの中から特にキーボードをセンスするような常駐プログラムを常駐させたまま終了させてしまうと、システムエラーを出してしまう。もっとも、常駐プログラムを使うときは、ユーザが登録と解除の順番に気を配らなければならないことは常識になっているし、Fuから起動されたプログラムは、Fuがなくなってしまえば戻る先がなくなってしまうわけなので、くれぐれも注意すること。また、メモリに不用意な不連続領域ができるので、メモリの無駄遣いにもなりかねないので、注意すること。

**【Fu支援ツール】**……　Fuは、Fuから起動され使用されることを前提とした、いくつかの支援ツールとでもいうべき外部コマンドが、ユーザの手によって製作され、公開されています。これらのFuの支援ツール類を用いることで、あたかもFuの仕様が拡張されたかのような、違和感のない統一された環境と画面で、使用することができるようになります。以下、筆者が知っている主な支援ツールを紹介します。

**＜プログラム名＞** FuTree.r
**＜作者＞** Tomさん

Fuのファイル画面はファイル一覧形式で表示されますが、“FuTree.r”は、ツリー形式でのファイル表示を行うツールです。Fuがツリー形式によるディレクトリ表示機能を持っているかのような使用感を実現しています。ツリー形式の表示は、ディレクトリの形を一覧できるという長所があります。

**＜プログラム名＞** Fudrive.r
**＜作者＞** Tomさん

Fuにドライブ移動のためのドライブ変更メニューを追加するツールです。DRIVE.Xのようにドライブの並びが確認できる他、カーソルキーでドライブを移動することが可能になります。現在では、Fuがこの機能を内蔵しています。

**<プログラム名> Fucon.X**

**<作者>** Emuさん

Fuは、画面下部に5行ほどのコンソールエリアを持ち、外部コマンドの出すエラーメッセージやヘルプ画面などをここに表示します。しかし、5行では、たとえば、LHA.xの圧縮ファイル一覧（“-l”オプション）などを使って多数のファイルが格納された圧縮ファイルを表示すると、表示内容が画面からあふれ、スクロールしてしまいます。これでは、せっかくの表示も正しくコマンドが実行されたのか、何かエラーが出たのか確認できません。そこで外部コマンド実行時にFuのメイン画面である、ファイル表示画面に一時的にコンソール表示を移そうという発想で公開されました。ただし、現在では、Fuがこの機能を外部コマンドの「オプション機能」として内蔵しています。

**<プログラム名> Fucust.X**

**<作者>** TFU PROJECT

Fuにメニュー機能を実現させるために公開されたツールです。現在のFuは、このメニュー機能を内蔵していますが、Fucust.Xはさすがに専用に製作されたツールだけあって、より優れた機能を持っています。1つのキーに26個のコマンドを最大4ページ用意しています。また、Fuがカーソル移動でメニューを選択するのに対して、“Fucust.X”は[A]～[Z]のキーに機能を割り当てているため、一発で起動することが可能です。

**【その他のファイルユーティリティ】**…………

X68000用のファイルユーティリティとしては、Fu以外にも数多くのユーティリティが公開されています。どれも、それぞれ独自の特徴を持ち、機能的には充実しています。その半面、ユーザが「他のファイラにあるのだから、同じような機能を実現して欲しい」などといった形で次々に要望を出してくるので、頻繁にバージョンアップしているツールはどれも基本機能は同じようなものになってきているようです。どれが優れているかなどの問題ではなく、純粋にユーザの好み（設定ファイルが理解しやすい、細かい環境設定部分で好みの環境が作れる、画面レイアウトが好きなど）でいろいろ選べる環境にあります。これは大変恵まれた環境だといえるでしょう。

以下に主なファイルユーティリティを紹介します。

**<プログラム名> TF.X**

**<作者>** Thuleさん、ippohさん、里衣座さん、AKTさん

製作者が次々と変わりながらバージョンアップを重ねているファイルユーティリティです。原作者はThuleさんで、以前「ASCII」誌上で紹介され、ソフトの自動販売機「TAKERU」でTAKERUの使用料のみという価格で販売されていました。画面レイアウトがシンプルで洗練され、応答性もよく、内蔵ビューワも優れているという素晴らしいユーティ

リティです。また、2画面に分割されたファイル表示画面は2つのディレクトリを参照しつつファイル操作ができるため、わかりやすく確実な操作を実現しています。

昨年まで、Thuleさんが多忙のため、バージョンアップが止まっていましたが、ippohさんがこれを引き継ぎ、さらに充実した内容になるとともに、TwentyOne.xやlndrv.xなどへの対応を実現しました。その後、さらに里衣座さん、AKTさんがバージョンアップを引き継ぎ、多彩な機能を充実させています。製作者がこれだけ引き継がれていくのも、TFが優れた思想と機能を持っているゆえ、ユーザがほうっておかないからでしょう。熱烈なユーザを多く持ったユーティリティです。

**<プログラム名>** DImain.x、DI.R
**<作者>** OOYAMAさん

これも熱烈なユーザを擁するユーティリティです。こちらは以前、「I/O」誌上で紹介されました。DIの特徴は、自由度の高い設定ファイルを記述できることと、応答性のよさでしょう。実際、使いこんだユーザは、個々人で異なったDIを使用しているようで、単に機能を割り当てたキーが違うという程度ではありません。また、メモリの使用量もそれほど多くありませんし、メイン部分をメモリに常駐させているため、高速な起動を可能にしています。

**<プログラム名>** MF.X
**<作者>** 雅さん

本書でも紹介されていますので、詳細はそちらのページを見ていただくとして、TFと同じく、2画面のファイル画面を持つファイルユーティリティです。TFのバージョンアップが中断していたときに、COMMAND.X以外のシェル、TwentyOne.xなどに対応したファイルユーティリティを目的に製作されました。大量のコマンドをキーに割り当てられるなど、かっちり製作された独特のイメージを持つファイルユーティリティです。

**<プログラム名>** FDX.X
**<作者>** ヒイナさん

Fuと同様に、NECのPC-9800シリーズで公開されているFD.EXEを参考に製作されたユーティリティです。最近開発が中断されているようですが、応答性もよく、これからが楽しみなユーティリティです。

**<プログラム名>** CDF.X
**<作者>** ASTさん

NECのPC-9801用のソフトで「エコロジー」というユーティリティがありますが、

CDF.Xは画面まわりはエコロジーと同様にツリー表示を基本にしています。X68000では、この形式のユーティリティは、少数なため、貴重な存在だといえます。

**<プログラム名> WS.X**

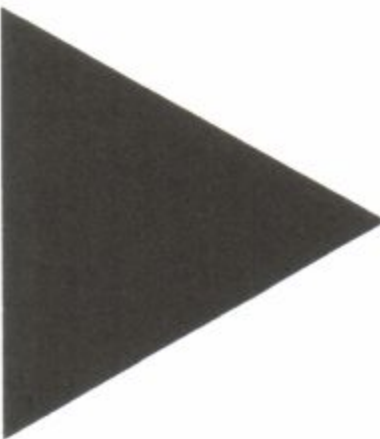
**<作者>** Fuchiさん

Fuを参考にして製作されたユーティリティです。TFなどのように2画面のファイル表示画面を持ち、画面を参照しながら作業をすることができます。基本機能は揃っていますし、さらに独自の機能の発展を遂げています。比較的後発ながら、ユーザを増やしているファイルユーティリティです。

**【最後に】**

筆者がX68000ユーザになって最初に出会ったユーティリティはTFでした。その後、設定ファイルの自由さを求めて(当時のTFは、今ほど自由になんでもコマンドを登録できるわけではありませんでした)、Fuに移りました。Fuの設定ファイルは、まだ通信を始めたばかりの初心者の私にもわかりやすく、設定しやすい設定ファイルでした。サンプルの設定ファイルでも一通りの機能は使用できますが、ファイルユーティリティは、自分なりにカスタマイズしてこそ、本当に自分にとって使いやすいものになるように思います。

Fuに添付のドキュメントは、はじめてFuに触れる方には簡潔すぎるかもしれません。Fuが開発されてから長い時間がたっているため、すでにFuのようなユーティリティを知っていることを前提にマニュアルが書かれている部分もあるからです。今回は、できるだけ詳しく動作を説明するように心掛けました。しかし、それでも行き届かない部分、わかりにくい部分があるかと思います。そんなときは、あなた自身で設定ファイルをいじってみてください。間違っても大丈夫です。Fuが設定時にエラーを出して教えてくれますし、間違えても、あなたのマシンが壊れることはありませんから。いろいろいじっているうちにきっとあなたも環境を整備することが楽しくなることでしょう。

まだX68000を購入したばかりの人、パソコン通信をしたことがない人は、標準で添付されてくるシステムディスクや、数本のソフトウェアしか持っていないかもしれません。Fuはその内蔵機能だけで十分ファイル操作が楽になり、あなたのX68000ライフを快適なものにしてくれることでしょう。しかし、Fuに限らず、ファイルユーティリティの多くは、なんらかの外部コマンドが起動することで、さらに便利な環境を提供してくれます。このことは、パソコン通信を通じて便利な外部コマンドが揃えば揃うほど、どんどんファイルユーティリティが便利になっていくことを示しています。筆者は一部のネットで嫌われているほどDOM、ROMを否定したくありません(用語は第1章を参照してください)。誰でも最初にパソコン通信を始めた頃は、便利そうなプログラムに心を奪われた経験が多かれ少なかれあったと思うからです。一通り自分の環境が整備されたあとで、どうなるかは別の問題ですが。とにかくFuというベースはここにあります。あとは自分自身で、さらに快適な環境作りを目指してください。

# MF.X

## ユーザが独自の環境を構築できる、柔軟性に富んだ2画面型ファイラ

【概要】……………………　このプログラムは、Human68kにおけるファイルおよびディレクトリの操作、プログラムの実行などをカーソルで選択することによって行うCOMMAND.X支援プログラムです。

その特徴としては、

(1) TwentyOneによりサポートされたマルチピリオドファイル（TwentyOneの解説ページ参照）に対応している
(2) すべての操作（基本コマンドも含む）を自分だけの使いやすい環境に設定できる
(3) ツリー表示をサポートしている
(4) 拡張子を判断してプログラムを実行することが可能である（たとえば、拡張子がTXTならばエディタが起動するようになっている）。また、拡張子ではなく、ファイル名による判断も可能である。たとえば、MAKEFILEならmake.xを起動するという具合である
(5) SUBST.X等でドライブを割り当てた場合の対応（詳しくいえば、ディレクトリマウントへの対応）を行っている
(6) lndrv.x等のシンボリックリンク[注]に対応している

などが挙げられます。

特に通常のプログラムではメニューの項目数とか形式とかが決められていますが、MF.Xではメニューの形式も各自が自分の使いやすいように設定できるようになっています。つまり、基本的にすべてのことが自分でカスタマイズ可能です。

注：シンボリックリンクとは、他のドライブやディレクトリ上にあるファイルを、シンボリックリンクファイルという特別なファイルを通して、あたかもカレントディレクトリに実体があるかのようにアクセスできるものをいう。

【作者】……………………　雅

| | |
|---|---|
| MAX BBS | 378 |
| Feel NET | 13 |
| ひるまーねっと | 8 |
| Cecile BBS | 24 |
| ALTA-NET | 573 |
| EYE・COM-NET | 683 |

**【作者からの言葉】**……

Bash[注]を使っていた私は、その結果、TwentyOneのマルチピリオドファイルを使用するはめになってしまいました。以前からX68000には優秀なファイラが存在していたのですが、そのときに、はたと気がついたら、マルチピリオドに対応しているファイラが見つからなかったのです。結局、マルチピリオドに対応したファイラということで作ったのがMF.Xです。

MF.Xを作るに際しては、操作性の向上と高機能化を中心に行いました。すでに発表してから1年以上たっていますので、よほど特殊なことを行うのでなければ、COMMAND.Xの存在を忘れさせてくれるでしょう。

私自身は、X68000を立ち上げたあとは、MF.X を起動すれば、あとはMFの上から通信は行うわ、プログラムは作るわと、メモリの許す限りのことを行っています。さあ、あなたもCOMMAND.Xを忘れましょう。

注：Bash ＝ GNU Bourne Again SHell（UNIXのBourne ShellのGNU版のX68000版）X68000のCOMMAND.Xのかわりに使用する。

**【推薦します】**…………

2面のドライブ表示は、ファイルの複写・移動をする際に視覚的な確認がしやすく、ツリー表示ではドライブ全体を容易に見渡すことが可能となり、重宝しています。また、MF.Xのキーアサインに関する自由度は特出したものがあって、MF.Xの持つほとんどすべての機能を好きな形で実行するように設定できるので、使うほどに自分にあったファイラへと成長することでしょう。

MAX BBS　いさお

**【起動方法】**……………

MFを起動させる前に、まず、MF.XとMF.DEFが同一ディレクトリにあることを確認しておいてください。

次に、コマンドラインから、

MF[CR]

と入力することにより起動します。

起動させたばかりのMF.Xにおいては、カーソルの移動(後述の標準キー)以外のキーは設定してありません。それ以外のキーは、すべてユーザが設定することにより、使いやすさが増していくわけです。キーボードの、どのキーを押したら、どのような動作を行うかは、MF.DEFに設定しておけば、MF.Xが解析して実行してくれます。

**【環境設定】**……………

環境変数の設定をする必要はありません。MF.Xを起動した場合にMF_BIN.DEFが存在していなければ、自動的にMF_BIN.DEFが作成されます。このMF_BIN.DEFは、MF.DEFをバイナリ変換して作成したもので、MF.Xの起動に必要なものが記録してあ

ります（通常、MF_BIN.DEFの存在は気にする必要はありません）。MF_BIN.DEFはMF.DEFが書き直された場合にも新たに作成されます。次の起動からは、このMF_BIN.DEFを読み込んで起動します。

**【使用法】**…………………　MFを立ち上げると、画面は次のようになります（今、ドライブA:にいる場合。もちろん、マシンによって表示されるファイル名等は違うとは思いますが）。

Pht.1　MFの起動画面

```
  MF.X G2.06-α  X680x0   Run-Time[     0: 0: 0]                                  土曜 1993-02-13 15:00:13
File=[   7] Directory=[   2]           MARK=[  0] File=[   7] Directory=[   2]           MARK=[  0]
File is [  279362Byte] MARK is [          0Byte] File is [  279362Byte] MARK is [          0Byte]
[ A:/                                           ] [ A:/                                           ]
•BIN                    <DIR>  93-02-13 11:33:28 •BIN                    <DIR>  93-02-13 11:33:28
 SYS                    <DIR>  93-02-13 11:40:36  SYS                    <DIR>  93-02-13 11:40:36
 autoexec.bat              82  93-02-13 12:20:20  autoexec.bat              82  93-02-13 12:20:20
 COMMAND.X              28312  92-02-11 12:00:00  COMMAND.X              28312  92-02-11 12:00:00
 config.sys               233  93-02-13 11:44:06  config.sys               233  93-02-13 11:44:06
 foo.txt                 1394  93-02-10 22:42:00  foo.txt                 1394  93-02-10 22:42:00
 MF.DEF                 35036  93-01-18 12:00:00  MF.DEF                 35036  93-01-18 12:00:00
 MF.X                  164998  93-01-18 12:00:00  MF.X                  164998  93-01-18 12:00:00
 MF_BIN.DEF             49307  93-02-13 11:46:36  MF_BIN.DEF             49307  93-02-13 11:46:36
 -.-.-.-.-.-.-.-End of Directory-.-.-.-.-.-.-.-   -.-.-.-.-.-.-.-End of Directory-.-.-.-.-.-.-.-

 Max=[  1250K] Use=[   440K] Free=[   809K]       Max=[  1250K] Use=[   440K] Free=[   809K]
```

## 1. ファイルの複写・削除

ここでファイルの複写を行う（ドライブA:にあるファイル名“foo.txt”をドライブB:に複写する）場合は、次のような操作で行います。

①[D]キーを押してドライブ選択表示画面を出す。そして、[↑][↓]キーでドライブB:を選択して[CR]キーを押す。すると、画面は、左側がドライブB:になり、右側はドライブA:になる。
②[→]キーでカーソルを右側のドライブA:に移動させる。
③[↑][↓]キーでカーソルをファイル“foo.txt”の場所に移動させる。
④[SPACE]キーを押す。すると、画面は、“foo.txt”の部分の色が反転した状態になる。
⑤ここで、[C]キーを押せば、“foo.txt”が複写される。

これで複写が完了です。画面の左側のドライブB:にファイル“foo.txt”がコピーされたことが確認できますね。①については、[CTRL]+[B]キーでも同じことが実行できます。

次に、ファイルの削除です。上で複写したドライブB:の“foo.txt”を削除してみましょう。カーソルキーを“foo.txt”にあわせ、⑤の[C]キーのかわりに[K]キーを押します。すると、画面には次のような表示が出ます。

**Pht.2** 削除確認の画面

```
 MF.X G2.06-α  X680x0  Run-Time[   0: 0: 0]                                  土曜 1993-02-13 15:01:24
File=[   2] Directory=[   3]          MARK=[  1] File=[   7] Directory=[   2]            MARK=[  0]
File is [     1411Byte] MARK is [       1394Byte] File is [   279362Byte] MARK is [          0Byte]
[ B:/                                            ] [ A:/                                           ]
 BIN                    <Dlink> 93-02-13 12:49:26  BIN                    <DIR>  93-02-13 11:53:28
 doc                    <DIR>   93-02-13 12:28:42  SYS                    <DIR>  93-02-13 11:40:56
 txt                    <DIR>   93-02-13 12:28:36  autoexec.bat              82 93-02-13 12:20:20
 autoexec.bat           -Flink- 93-02-13 12:49:28  COMMAND.X              28312 92-02-11 12:00:00
*foo.txt                   1394 93-02-10 22:42:00  config.sys               233 93-02-13 11:44:06
 -.-.-.-.-.-.-.-End of Directory-.-.-.-.-.-.-.-.  foo.txt                 1394 93-02-10 22:42:00
                 delete B:/foo.txt  Yes or No or Allyes                   35036 93-01-18 12:00:00
                                                   MF.X                  164998 93-01-18 12:00:00
                                                   MF_BIN.DEF             49307 93-02-13 11:46:36
                                                   -.-.-.-.-.-.-.-End of Directory-.-.-.-.-.-.-.-.

 Max=[  1250K] Use=[     6K] Free=[  1244K]      Max=[  1250K] Use=[   440K] Free=[   809K]
copy [A:¥foo.txt] ===> [B:¥foo.txt] Ok !!
delete  [B:¥foo.txt]
```

ここで[Y]キーを押すと、“foo.txt”を削除することができます。もし間違って別のファイルを指定してしまった場合は、[N]または[ESC]キーを押すことにより、削除を中止することができます。

また、ファイルの移動の場合は、⑤で[C]キーのかわりに[SHIFT]+[C]キーを押すだけで、他は複写の場合と同じです。ただし、最後に削除の場合と同じ確認の表示が出ます。これは、ファイルを移動する場合は、移動先が同一ドライブでない場合には、複写を行ったあとに削除を行うためです。もちろん、同一ドライブの場合にはその表示は出ません。

ここでは、“foo.txt”という１つのファイルを例にとり、複写・削除・移動の実際を説明しましたが、複数ファイルに対して行うには、④の[SPACE]キーでのマークを複数ファイルに対して行えば一括して行えます。

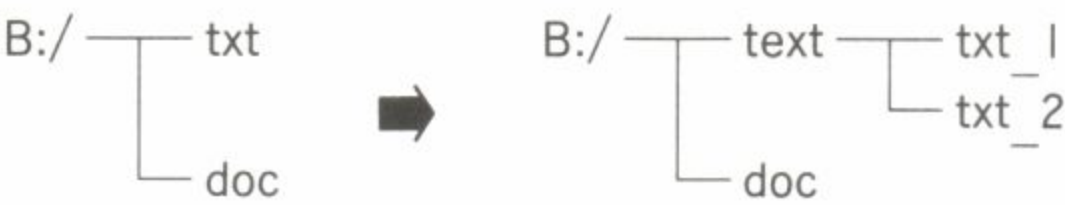
また、１つのファイルに対してのみ行う場合は、マークをせずに、そのファイルにカーソルを移動して[C](複写)、[K](削除)、[SHIFT]+[C](移動)キーを押すだけでも可能です。

## 2．ディレクトリの作成

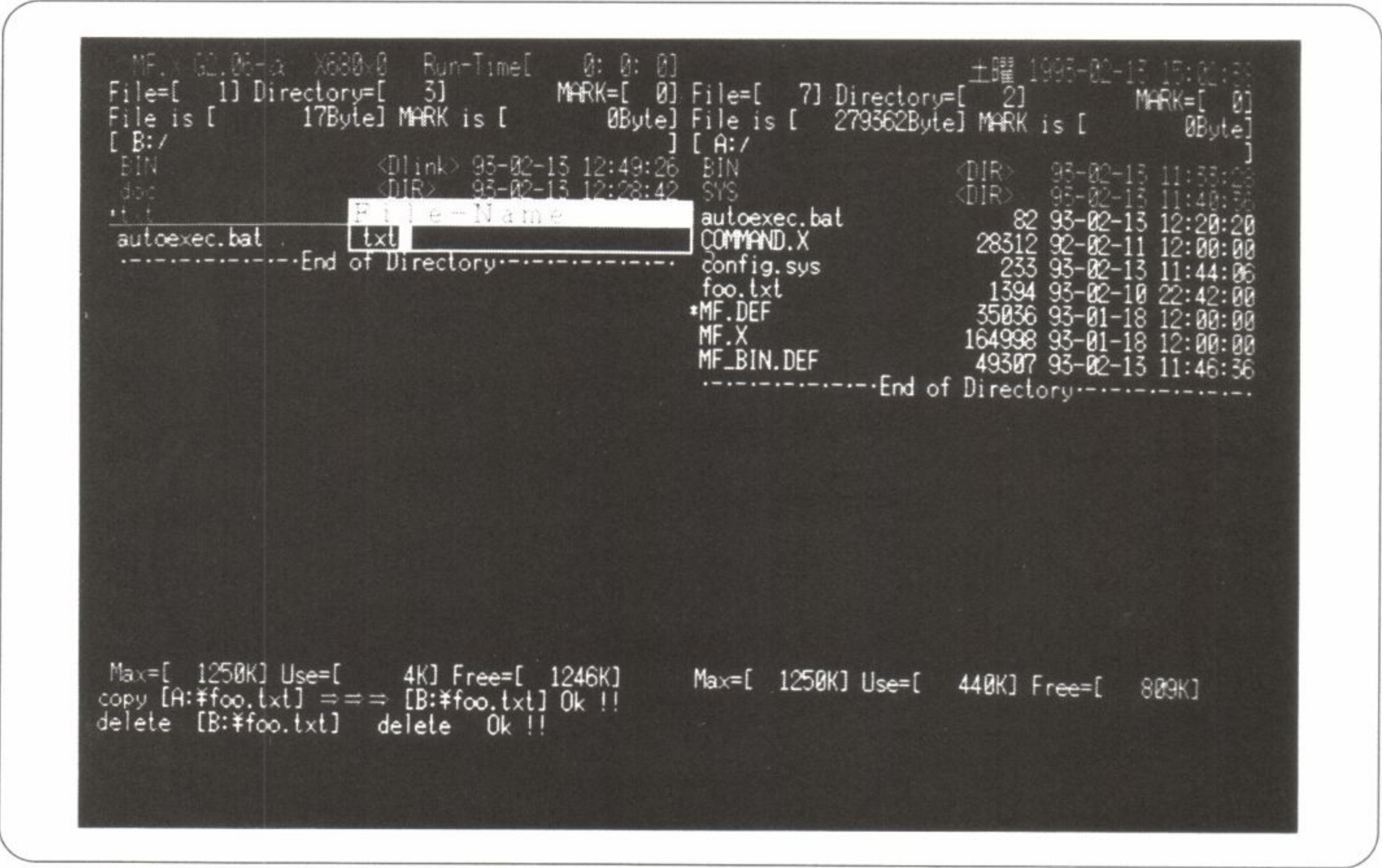
次にディレクトリの作成を行う場合を考えてみましょう。ドライブB:のディレクトリを次のように作り変えるとします。

B:/ ─┬─ txt
　　 └─ doc

➡

B:/ ─┬─ text ─┬─ txt_1
　　 │　　　　 └─ txt_2
　　 └─ doc

① カーソルをディレクトリ“txt”に移動する。

② [R]キーを押すと、画面に次のような表示が出る。

Pht.3　リネーム入力画面

③ [→][←]キーを使用して“x”の文字の上にカーソルを持っていき、[e]を押す（その他、[DEL][BS][INS]キーなどがコマンドラインでのHISTORY.X上での動作と同様に使える。これらの制御キーは、MF.DEFのPa-A（後述）で変更または拡張することが可能）。

④ [CR]キーを押す。これで“txt”ディレクトリがディレクトリ“text”に変わる。（ディレクトリ名の変更）

⑤ 今、カーソルはディレクトリ“text”の場所にあるので、[CR]キーを押せばサブディレクトリ“text”に下りられる。間違って、ディレクトリ“doc”の中に入ってしまった場合は、[UNDO]キーを押せば元の上のディレクトリに戻れる（また、[↑]キーを押してカーソルを一番上の行（Top of Directory）に持っていき[CR]キーを押せば、上のディレクトリに行くことも可。本来、こちらのほうが正式な動作である）。

⑥ [M]キーを押すと、画面には次のような表示が出るので、

Pht.4　ディレクトリ名入力画面

⑦ [t][x][t][_][1] と押して新しいディレクトリ名“txt_1”を入力する。

⑧ そして、[CR]キーを押せば“txt_1”が作成される（ディレクトリの作成）。⑦での入力を間違えた場合は、[CR]キーを押す前にカーソルキーか[DEL][BS][INS]キーで修正するか、[CR]キーを押したあと、もう一度[R]キーを押して入力し直す。

同様にして、“txt_2” も作ってください。

## 3．外部プログラムの起動

さて、ここまでの説明で、あなたはMF.X上でのドライブ移動とディレクトリの移動ができるはずです。それでは、“CHKDSK.X” を動かしてみましょう。MF.Xはディレクトリ管理の他にシェルの役目も果たしますので、外部プログラムを稼働させることもできます（本来はこちらの機能のほうが重要であり、メモリを思う存分使えるX68000のうま味を感じることができます）。

① [D][UNDO][↑][↓][→][←]を使って、ドライブ、ディレクトリを変え、カーソルを“CHKDSK.X” の上に持っていく。

② ここで[CR]キーを押す。

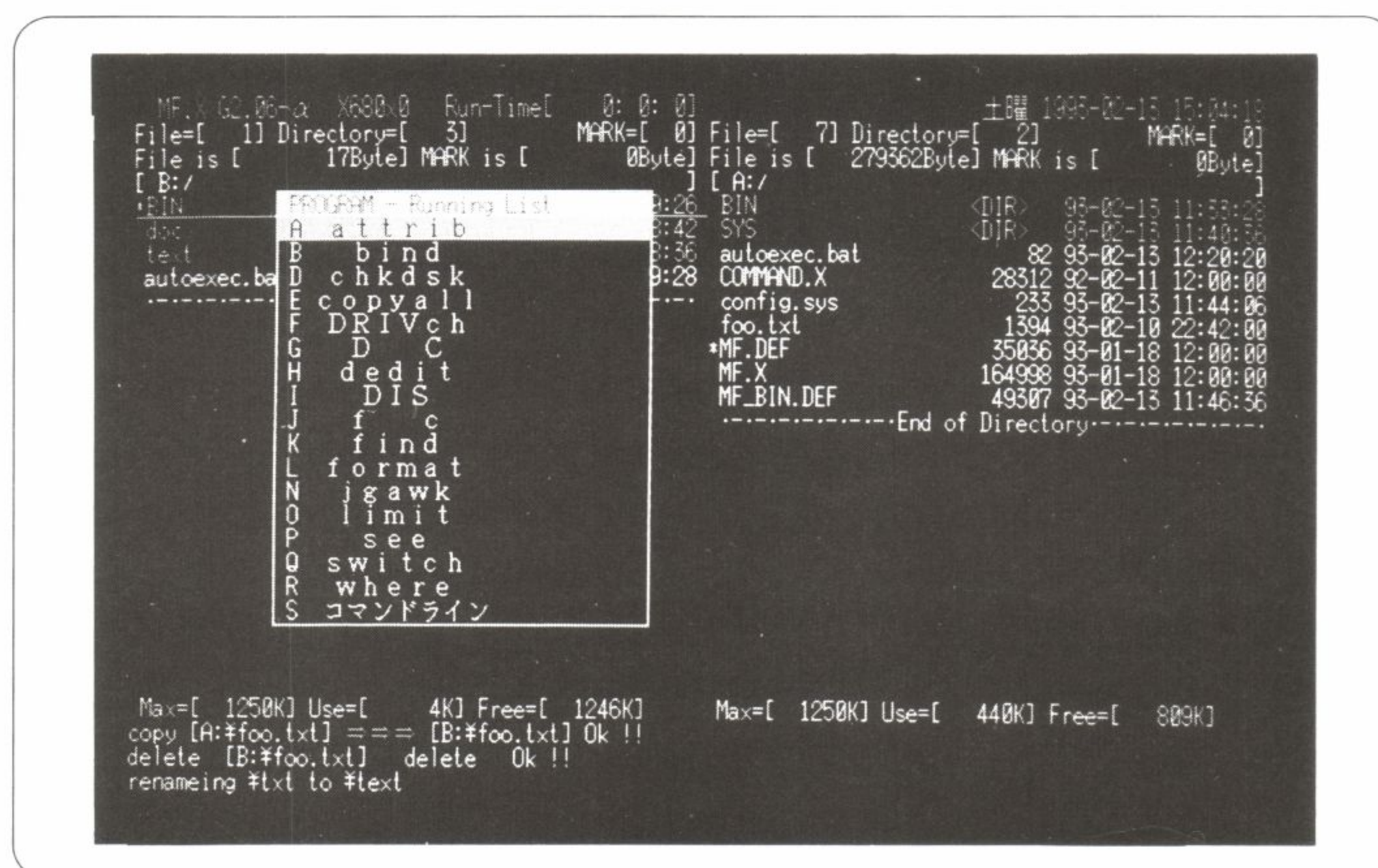
③ 画面に“Option”と表示されるが、そのままもう１度、[CR]キーを押す。すると、“CHKDSK.X” が実行される。

これだけでCHKDSK.Xが実行できました。簡単でしょう。

“CHKDSK.X”などのように、よく知っているプログラムの場合は、MF.Xでドライブの変更とかディレクトリの変更とかを行って実行するよりも、コマンドラインから入力したほうが早いかもしれません。しかし、通信を始めると、この本に収録されているようなプログラムがハードディスクに入りきらないくらいアップロードされているのに出合うでしょう。半数以上は覚えきれないようなプログラム名でしょうから、こういうことができる便利さがすぐにおわかりいただけるでしょう。

ちなみに、今、実行した“CHKDSK.X”は、次のように操作しても実行可能です。

① [X]キーを押すと、画面に次のようなメニュー表示が出る。

Pht.5　メニュー画面

② [↑][↓]キーを使って“D chkdsk”を指定する。

③ [CR]キーを押す。

これならコマンドラインから入力するよりも早く操作できます(これは、MF.DEFのメニューの項目に登録されているから可能なのですが)。

以上、MF.Xの機能の中から3つの機能を紹介しました。この調子で紹介を続けると、紙数を大幅に超えてしまいますので、以下、オリジナルのMF.DEFに登録されたキー割り当てを紹介しますので、自分で実際に操作してみてください。使いにくい部分は、自分でMF.DEFを書き換えてください。MF.DEFの書き換え方については、このあとで説明します。

## 【標準キー】……………オリジナルMF.DEFによる設定キー

次のものは、標準でアーカイブされているMF.DEFに設定してあるキーです。なお、説明にあるPa-?は、MF.DEFの動作を設定するものです。詳しくは、このあとの説明をお読みください。

### ●ディレクトリ・ファイル操作に関するもの

| キー | 内　容 |
|---|---|
| [C] | ファイル複写（MF.DEF内の#COPYを参照） |
| [SHIFT]＋[C] | ファイル移動（MF.DEF内の#MOVEを参照） |
| [K] | ファイル削除（MF.DEF内の#KILLSを参照） |
| [M] | ディレクトリ作成（#MKDIRを参照） |
| [R] | リネイム（#R-NAMEを参照） |
| [Z] | ファイル属性変更（#RE-ATRを参照） |
| [F] | ファイル名検索（#FIND-FNを参照） |
| [A] | すべてのファイルをマークする |
| [HOME] | すべてのファイルのマークを解除する |
| [CLR] | ディレクトリの再読み込みを行う |

### ●ドライブ・ディレクトリ移動に関するもの

| キー | 内　容 |
|---|---|
| [D] | ドライブ変更（#CHDRIVEを参照） |
| [DEL] | ルートディレクトリに移動する |
| [UNDO] | ディレクトリを１つ上に移動する |
| [CTRL]＋[A]～[CTRL]＋[J] | ドライブA:～J:に移動する |

### ●MF.Xそのものの環境に関するもの

| キー | 内　容 |
|---|---|
| [ESC] | MFを終了する |
| [I] | 上部に情報を表示する（#HELP-INFを参照） |
| [SHIFT]＋[I] | 下部にドライブの容量等の情報を表示する（#HELP-DRVを参照） |
| [1] | FD0をイジェクトする |
| [2] | FD1をイジェクトする |
| [T] | ツリー表示（#TREEを参照） |
| [@] | MF.XのPa-0の項目の一時変更 |
| [HELP] | 基本コマンドの一覧表を表示する |
| [登録] | 著作権の表示を行う |
| [SHIFT]＋[OPT 1]＋[UNDO] | キーボードをロックする |

### ●子プロセス起動に関するもの（その１）

| キー | 内　容 |
|---|---|
| [L] | SEE.Xを起動する |
| [F1] | TMNを起動する |
| [¥] | COMMAND.Xを起動する |

**●子プロセス起動に関するもの(その2)**

| キー | 内　容 |
|---|---|
| [E] | Pa-3(A) では次の拡張子が設定されている<br>.C　エディタを起動（即時実行）<br>.H　エディタを起動（即時実行）<br>上記以外の拡張子の場合、エディタを起動する直前に確認を求める |
| [O] | Pa-3(B) では次の拡張子が設定されている<br>.LZH　LZH展開<br>.ISH　ISH展開<br>.LOG　ISH展開<br>.X　LZX展開<br>.BFD　Bup展開 |
| [SHIFT]+[CR] | Pa-3(D) では次の拡張子が設定されている<br>.X　実行<br>.R　実行<br>.BAT　COMMAND.Xを介して実行<br>上記以外の拡張子の場合、パラメータを指定して実行 |

**●子プロセス起動に関するもの(その3)**

| キー | 内　容 |
|---|---|
| [J] | Pa-4(A)では、次のものが設定されている<br>指定したディレクトリに移動する |
| [S] | Pa-4(B)では、次のものが設定されている<br>ソートの項目を指定する |
| [U] | Pa-4(C)では、次のものが設定されている<br>圧縮処理を行う<br>LHaにより圧縮<br>LHaにより非圧縮<br>LHにより圧縮<br>Bdifにより差分抽出<br>LZXにより圧縮<br>ishにより変換<br>tarにより変換 |
| [W] | Pa-4(D)では次のものが設定されている<br>表示するものをワイルドカードで指定する |
| [X] | Pa-4(E)では次のものが設定されている<br>以下の各種プログラムを起動する<br>ATTRIB、BIND、CHKDSK、COPYALL、DC、dedit、DIS、FC、find、FORMAT、jgawk、limit、SEE、SWITCH、where |

**【カスタマイズ】**……… 最初に紹介したように、MFはユーザが自分の使いやすいようにカスタマイズできる柔軟性が特徴といえます。このカスタマイズする内容を記述するのがMF.DEFです。MF.DEFは、通常のテキストファイルです。エディタ等で編集してください。ユーザは、このMF.DEFを自分なりに書き直していくわけです。MF.DEFの内容は、大きくPa-0、Pa-1、Pa-2、Pa-3、Pa-4、Pa-Aの6種類に分けられます。それぞれの関係は、設定する内容に応じて階層化されています。つまり、Pa-0にはMF.Xの全体的な操作環境を、Pa-1、Pa-2、Pa-3、Pa-4には子プロセスで実行するプログラムおよび基本コマンドを、Pa-Aにはコンソール入力状態における[CTRL]キーの動作状況を、それぞれ設定していき

ます。この設定の内容により、階層化メニューの実現、拡張子（ファイル名、ディレクトリ名）による判断・実行の実現が可能です。

つまり、まとめれば、

| 設定ファイル | 内　容 |
|---|---|
| Pa-0 | MFの基本環境（キーリピートの間隔等）の設定 |
| Pa-1 | 実行（操作）したいファイル（プログラム）にカーソルを移動して、[CR]キーを押すことによって拡張子を判断し、実行させる設定をする |
| Pa-2 | キーボードのキーにより、どのような動作(たとえば、[E]によりエディタを起動する等）を行わせるかの設定をする |
| Pa-3 | Pa-2によって呼び出される設定<br>Pa-2で設定したキーで、Pa-1と同様の拡張子による動作を設定する |
| Pa-4 | Pa-2によって呼び出される設定<br>Pa-2で指定したキーで窓を開いて選択させること（プログラムの実行等）を設定する |
| Pa-A | コンソール入力状態におけるコントロール制御記号([CTRL]＋[L]等)の動作を設定する |

ということになります。

以下、それぞれの書き方を簡単に説明しておきます。

**●例１　起動するエディタの指定＋拡張子の指定＋読み込むファイル数を指定する場合**

① 目的のファイルにカーソルを移動し、[CR]キーでエディタ“ED.X”を起動させる場合。

② エディタを起動する場合は、カーソルが拡張子“.TXT”のファイルの場所にある場合のみとする。

③ そのとき読み込むファイルはあらかじめマークされているファイルで、一度に読み込むファイル数（最大19個）は６とする（ただし、マークされているファイルがない場合は、カーソルのある場所のファイルを読み込むことになる）。

この場合は、ファイル名の拡張子が“.TXT”であることと、キー入力が[CR]であることという２つの条件によってプログラム（エディタ）が起動するので、“Pa-1”に設定します。

まず、ファイル名の拡張子が“.TXT”で起動するには、

```
.TXT
```

とします。

次に、起動するプログラムが“ED.X”なので、

```
.TXT   ED.X
```

とエディタ名を指定しておきます。そのときにファイルを読み込むようにするので、

```
.TXT   ED.X %f
```

とします。
　読み込むファイル数を“6”とするので、

```
.TXT   ED.X %f6
```

とします。なお、%f数字は、数字で指定した個数までマークしたファイルを展開します。マークが行われていないと、カーソルが指しているファイルを展開します。
　で、最後に終了のサインを書き込んで、

```
.TXT   ED.X %f6  @@
```

となります。
　ただし、このままではプログラムの実行画面が下半分になってしまいます（これはMF.Xのデフォルトの動作です)。エディタの場合は全画面を使用しますので、その指定をする必要があります。指定方法は、実行するプログラムの前にオプションスイッチ“-d”を挿入します。

```
.TXT   -d ED.X %f6  @@
```

　これが、最終状態です。

**●例2　起動キーを指定する場合**

① [E]キーでエディタ“ED.X”を起動させる場合。
② そのとき読み込むファイルはあらかじめマークされているファイルで、1度に読み込むファイル数は6とする(ただし、マークされているファイルがない場合は、カーソルのある場所のファイルを読み込む)。

この場合は、キー入力が［E］キーという条件のみによりプログラム(エディタ)が起動

するので、“Pa-2” に設定します。

まず、入力するキーを指定します。

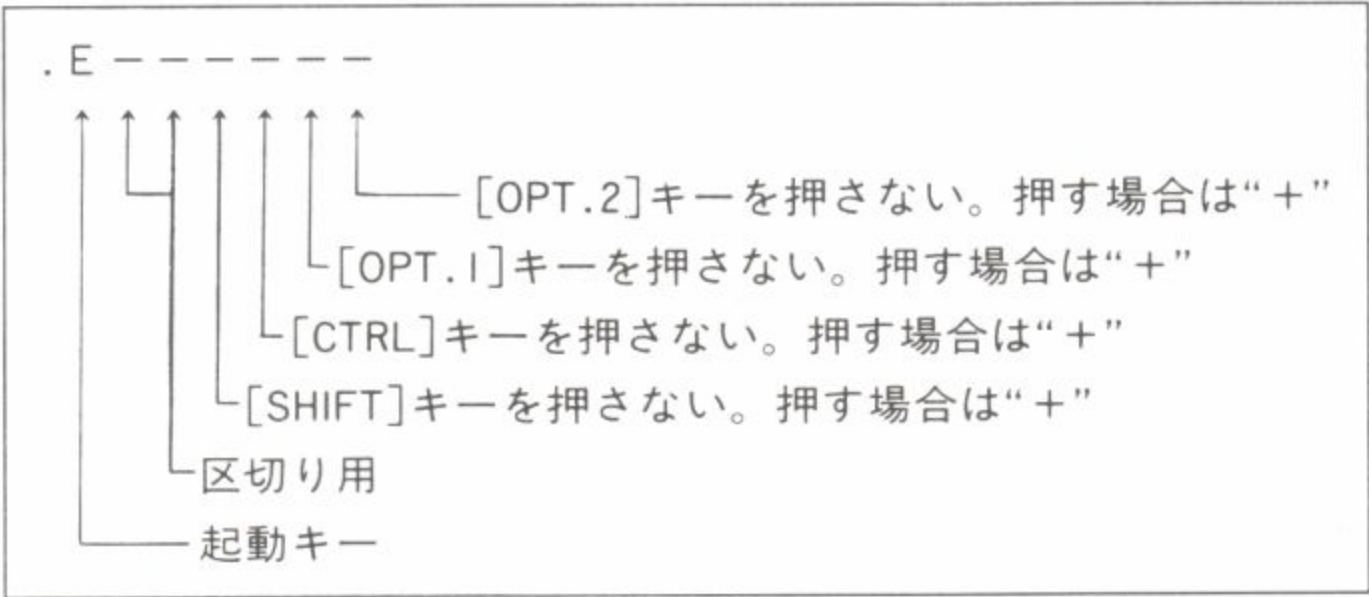
.E

次に、そのキーと他のキーの同時入力（MF.Xでは、通常のキーと同時に[CTRL][SHIFT][OPT.1][OPT.2]キーを押した場合、異なった動作を行わせることが可能)の状態を定義します。ここでは、[E]キーのみを押した場合の設定をします。

.E------

と設定します。“.E”以降の“-”には、それぞれ意味があります。3桁目と4桁目は5桁目以降の指定との単なる区切りです。ここで重要なのは先頭からの桁数です。5桁目は[SHIFT]キーを同時に押すかどうかの設定です、[E]キーのみを押すのであれば、“-”のままにしておいてください。[SHIFT]キーを同時に押すような設定にする場合は、“-” のかわりに“+”としてください。同様に、6桁目は[CTRL]キー、7桁目は[OPT.1]キー、8桁目は[OPT.2]キーを押すかどうかの指定ですので、それぞれ設定してください。

Fig.1 桁指定の意味

```
.E ------
 ↑ ↑↑ ↑ ↑ ↑ ↑
 │ └┘ │ │ │ └── [OPT.2]キーを押さない。押す場合は“+”
 │  │  │ │ └[OPT.1]キーを押さない。押す場合は“+”
 │  │  │ └[CTRL]キーを押さない。押す場合は“+”
 │  │  └[SHIFT]キーを押さない。押す場合は“+”
 │  └区切り用
 └──起動キー
```

.E------

次に、起動するプログラムが “ED.X” なので、

.E------ ED.X

とします。そのときにファイルを読み込むようにするので、

.E------ ED.X %f

とします。読み込むファイル数を“6”とするので、

```
.E------ ED.X %f6
```

とします。最後に終了のサインを書き込んで、

```
.E------ ED.X %f6 @@
```

となります。
　例1と同様に、全画面を使うので、

```
.E------ -d ED.X %f6 @@
```

とします。これが、最終状態です。

**●例3　複数の拡張子を指定してエディタを起動させる場合**

① [E]キーでエディタ“ED.X”を起動させる場合。
② エディタを起動する場合は、カーソルが“.TXT”、“.DOC”、“.TEX”、“.C”、“.S”の場所にある場合とする。
③ そのとき読み込むファイルはあらかじめマークされているファイルで、一度に読み込むファイル数は6とする(ただし、マークされているファイルがない場合は、カーソルのある場所のファイルを読み込むことになる)。

　この場合は、起動する条件がキー入力が［E］という条件の場合のみプログラム(エディタ)が起動するので、“Pa-2”に設定します。
　まず、入力するキーを指定します。例2の場合と同様に、キーの同時入力の状態を定義します。

```
.E------
```

　次に、起動する条件に拡張子の判断が加わっているので、“Pa-3”を使用して設定します。

```
.E------ Pa-3
```

“Pa-2”と“Pa-3”の接続コードを “A” とします。接続コードとは、“Pa-3” を呼び出すために必要な見出しと思ってください。使用可能なものは、AからZまでの大文字の英字です。指定する場合は、“Pa-3(A)”という具合に接続コードの英字の前後に半角の括弧が必要です。

```
.E------ Pa-3(A)
```

そして、最後に終了のサインを書き込んで、

```
.E------ Pa-3(A) @@
```

となります。これで、“Pa-2”の設定は終了です。

次に、呼び出される“Pa-3” を設定します。“Pa-3”の設定方法は、例1で設定した “Pa-1” と同様の設定方法です。

まず、どの拡張子で起動するかを指定します。

```
.TXT
.DOC
.TEX
.C
.S
```

次に、起動するプログラムが “ED.X” なので、

```
.TXT   ED.X
.DOC   ED.X
.TEX   ED.X
.C     ED.X
.S     ED.X
```

とします。そのときにファイルを読み込むようにするので、

```
.TXT   ED.X %f
.DOC   ED.X %f
.TEX   ED.X %f
```

```
.C      ED.X %f
.S      ED.X %f
```

とします。読み込むファイル数を“6”とするので、

```
.TXT   ED.X %f6
.DOC   ED.X %f6
.TEX   ED.X %f6
.C     ED.X %f6
.S     ED.X %f6
```

最後に終了のサインと画面の使用方法を書き込んで、

```
.TXT   -d ED.X %f6  @@
.DOC   -d ED.X %f6  @@
.TEX   -d ED.X %f6  @@
.C     -d ED.X %f6  @@
.S     -d ED.X %f6  @@
```

となります。

“Pa-2”と“Pa-3”の接続コードを“A”としたので、先頭行に!A、最終行に“.SUB-END”と書き込みます。

```
!A[CR]
.TXT   -d ED.X %f6  @@
.DOC   -d ED.X %f6  @@
.TEX   -d ED.X %f6  @@
.C     -d ED.X %f6  @@
.S     -d ED.X %f6  @@
.SUB-END[CR]
```

先頭行の“!A”と最終行の“.SUB-END”により、“Pa-3”ができあがります。このとき、“.SUB-END”が書き込まれていないと、その次に設定されたものとの区別がつかなくなりますので、注意してください。

**●例 4　メニュー形式から起動させる場合**

① [U]キーでアーカイバ各種を起動させる場合。
② アーカイバの種類は、メニュー形式でカーソルによって選択することにより起動させる。

この場合は、[U]キー入力のみでプログラム（アーカイバ各種）が起動するので、“Pa-2” に設定します。
まず、例2の場合と同様にキーの同時入力の状態を定義します。

```
.U------
```

起動する際にメニュー形式の中からカーソルキーで選択するようにしますので、“Pa-4”を使って設定します。

```
.U------ Pa-4
```

“Pa-2”と“Pa-4”の接続コードを“Pa-3”とは別に取り扱いますので、“C” とします。この接続コードも“Pa-3”と同じ意味のものです。“Pa-3”でAを使用している場合でも、“Pa-4”でまたAを使うことは可能です。

```
.U------ Pa-4(C)
```

最後に終了のサインを書き込んで、

```
.U------ Pa-4(C) @@
```

とすれば、これで、“Pa-2”の設定は終了です。
次に、“Pa-4”を設定します。
まず、メニュー項目の設定をします。項目ごとに、“.A”～“.E”を先頭から書いていきます。

```
.A
.B
.C
```

```
.D
.E
```

次に、起動するプログラムと、そのときのオプション、読み込むファイル等を設定していきます。

```
.A  -rc   LHa a -a1  %im  %f5  @@
.B  -rc   LHa a -z1  %im  %f5  @@
.C  -rc   LH -ah     %im  %f5  @@
.D  -krc  BDIF       %df  %f1  @@
.E  -rc   LZX        %f1       @@
```

なお、“-rc”、“%im”、“-krc”については、後述する「画面等の制御コマンド」で説明しますので、そちらを参照してください。

最後に選択する際のコメントを書き込みます。

```
.A   $LHa圧縮           $ -rc   LHa a -a1 %im  %f5  @@
.B   $LHa非圧縮         $ -rc   LHa a -z1 %im  %f5  @@
.C   $LH圧縮            $ -rc   LH -ah    %im  %f5  @@
.D   $Bdif  O-you N-my  $ -krc  BDIF      %df  %f1  @@
.E   $LZX圧縮           $ -rc   LZX       %f1       @@
```

また、“Pa-2”と“Pa-4”の接続コードを“C”としたので、先頭行と最終行に“!C”と“.SUB-END”をつけます。

```
!C[CR]
.A   $LHa圧縮           $ -rc   LHa a -a1 %im  %f5  @@
.B   $LHa非圧縮         $ -rc   LHa a -z1 %im  %f5  @@
.C   $LH圧縮            $ -rc   LH -ah    %im  %f5  @@
.D   $Bdif  O-you N-my  $ -krc  BDIF      %df  %f1  @@
.E   $LZX圧縮           $ -rc   LZX       %f1       @@
.SUB-END
```

これで、できあがりです。

もう1度、まとめてみましょう。

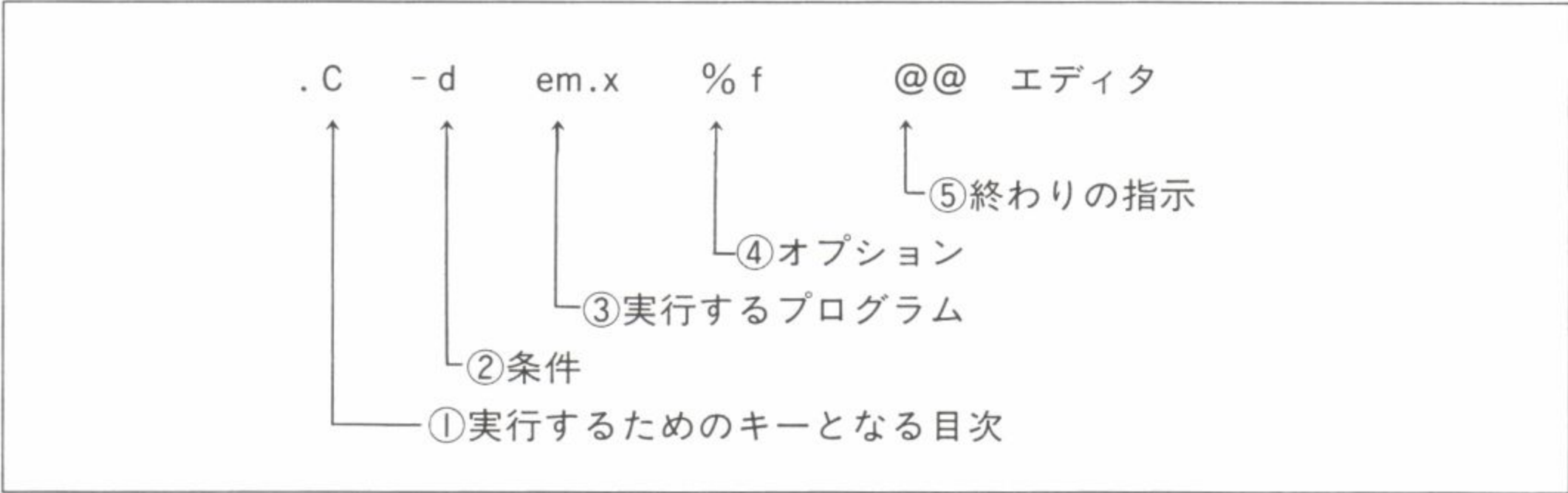
MF.DEFでは、次の例のように5個の部分を書いていくことになります。

| ① | ② | ③ | ④ | ⑤ |
|---|---|---|---|---|
| .C | -d | em.x | %f | @@　エディタ |

①は実行するためのキーとなる目次のようなもので、②はそのプログラムを実行する場合の条件等の設定です。次の③で、実行するプログラムを指定します。④はプログラムに渡すオプションを指定します。基本的には、この③と④は、通常のコマンドラインで入力するとおりに書いておけば問題はありません、ただし、④は決まった文字(先頭が%で始まっている文字)を書くことにより、より柔軟性に富んだ指定が可能です。⑤は終わりの指示と、見やすくするためのコメントです。

②から⑤までは“Pa-1”から“Pa-4”まで共通で、①だけが違った指定方法となっています。

Fig.2　MF.DEFの指定のしかた

```
.C   -d   em.x   %f     @@  エディタ
↑    ↑    ↑      ↑      ↑
|    |    |      |      └⑤終わりの指示
|    |    |      └④オプション
|    |    └③実行するプログラム
|    └②条件
└──①実行するためのキーとなる目次
```

ここでは、MF.DEFの設定項目のうち、最も参照する頻度の高い、②で指定する、「画面等の制御コマンド」についてだけ紹介しておきます。実際にMF.DEFを記述する場合は、この他にもいろいろな記述のしかたやルールがあります。それらについては、MF_X_206.Lzhにアーカイブされているドキュメントファイル MF_MAN.LZHを参照してください。

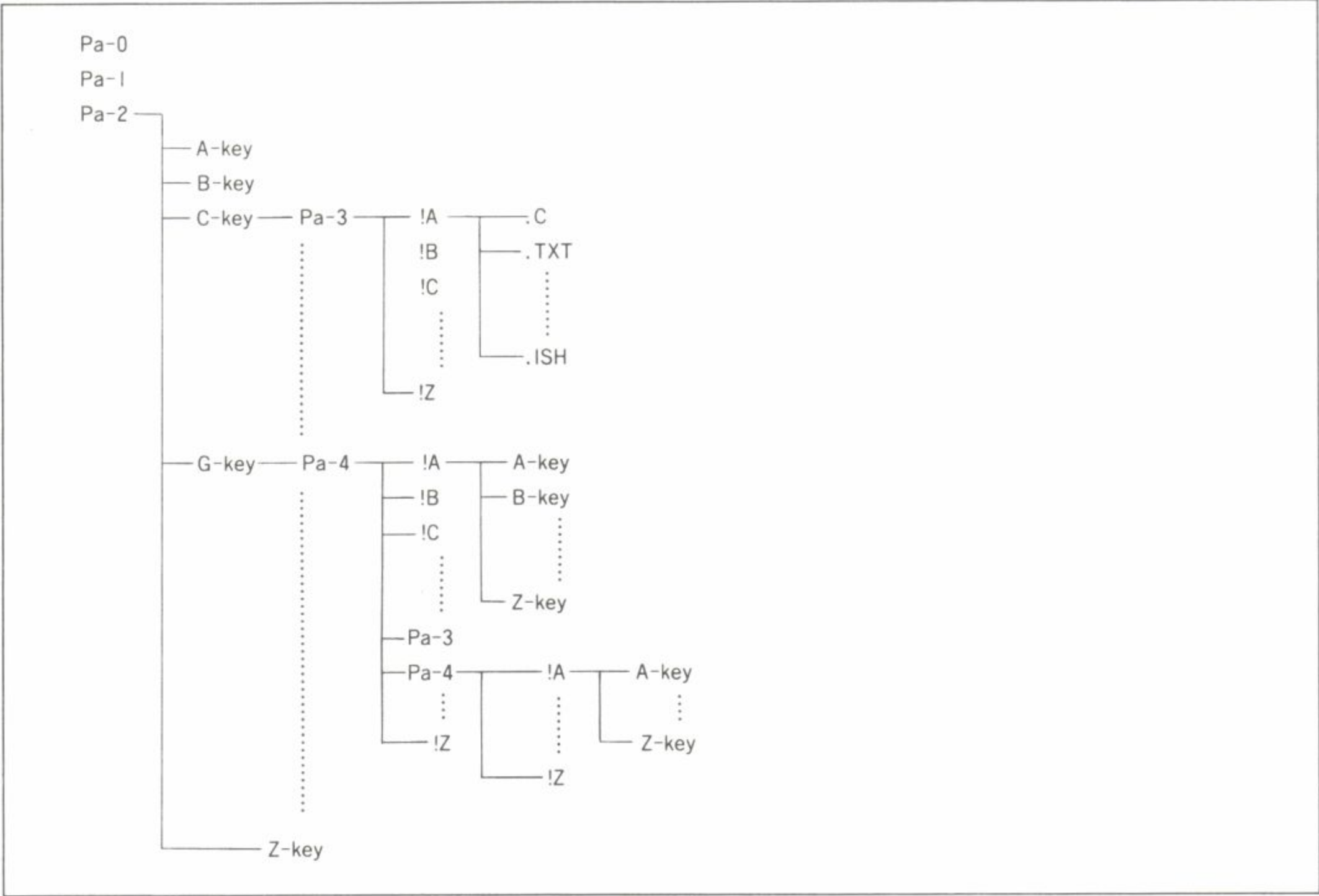
**●画面等の制御コマンド（オプションスイッチ）**

| オプションスイッチ | 内　容 |
|---|---|
| -d | プログラムを画面全体で行うようにする |
| -k | プログラムの終了時点でキーの入力を待つ |
| -r | プログラムの実行時点でキーの入力を待つ |
| -n | プログラムの終了後、ディレクトリの読み込みを行わない |
| -c | プログラムの終了後、画面の初期化を行わない |
| -p | プログラムの実行時点でパレットを初期化する |
| -P | プログラムの実行時点でパレットを初期化しない |

| | |
|---|---|
| -w | チャイルドプロセスの環境を継承する |
| -@ | 展開したコマンドラインを編集する |
| %im | コンソールからの入力をカーソルのあるディレクトリに追加して展開する |

なお、参考までに、Pa-0、Pa-1、Pa-2、Pa-3、Pa-4、Pa-Aの相互の関係がつかめるように、その関係の概要図を示しておきます。

**Fig.3**　MF.DEFの概要図

```
Pa-0
Pa-1
Pa-2 ─┐
      ├─ A-key
      ├─ B-key
      ├─ C-key ── Pa-3 ──┬─ !A ──┬─ .C
      │            :     │  !B   ├─ .TXT
      │            :     │  !C   │   :
      │            :     │   :   └─ .ISH
      │            :     └─ !Z
      │            :
      ├─ G-key ── Pa-4 ──┬─ !A ──┬─ A-key
      │            :     ├─ !B   ├─ B-key
      │            :     ├─ !C   │   :
      │            :     │   :   └─ Z-key
      │            :     ├─ Pa-3
      │            :     ├─ Pa-4 ──┬─ !A ──┬─ A-key
      │            :     │   :     │   :   │   :
      │            :     └─ !Z     │   :   └─ Z-key
      │            :               └─ !Z
      └────── Z-key
```

このようにして、MF.DEFを自分の環境に見合ったものに作り変えることによって、ゲーム以外のほとんどのX68000における作業は、コマンドラインに戻ることなく、MF.Xの上で可能です。設定のしかたによっては無限に近い組み合わせが可能です。

### ●MF.DEFで変更できないもの

なお、以下のキーは、MF.DEFで変更できないものです。これと同様なコマンドは、後述する基本コマンドの中にもあります。

| コマンド | 内　容 |
|---|---|
| [CR] | MF.DEF（Pa-1）で指定したとおりに実行してくれる<br>ちなみに、ディレクトリのところで[CR]キーを押すと、ディレクトリの変更をする（ただし、ツリー状態では変更はできない） |
| [←] | 画面を左へ移動する |
| [→] | 画面を右へ移動する |
| [↑] | カーソルを上へ移動する |
| [↓] | カーソルを下へ移動する |
| [ROLL UP] | [↓]の拡張（1画面スクロールアップ） |
| [ROLL DOWN] | [↑]の拡張（1画面スクロールダウン） |

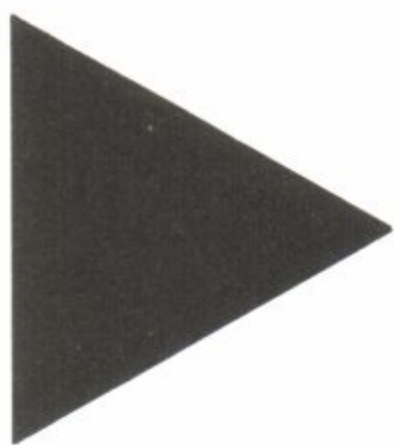
| | |
|---|---|
| [ESC] | コマンドの中止 |
| [SPACE] | ファイルのマーク |

**【内部コマンド】**……… なお、MF.Xでは、以下のような内部コマンドを使うことができます。これらの内部コマンドを組み合わせ、自分の使いやすいMF.Xを作ってください。

| 内部コマンド名 | 内　容 |
|---|---|
| #COPY | カーソルラインの表示されているドライブ(またはディレクトリ)から、反対の(“右”にカーソルラインがある場合は“左”の)ドライブ(またはディレクトリ。ツリー表示状態においてはアスタリスクの指示している場所)にファイルの複写を行う |
| #MOVE | カーソルラインの表示されているドライブ(またはディレクトリ)から、反対の(“右”にカーソルラインがある場合は“左”の)ドライブ(またはディレクトリ。ツリー表示状態においてはアスタリスクの指示している場所)にファイルの移動を行う |
| #KILLS | ファイルおよびディレクトリの削除を行う。実行すると、“Yes/No/All Yes”と聞いてくるので、頭文字の[Y]、[N]、[A]キーを押す。ツリー表示状態においては、ファイルのバイト数を 0 にして表示する。[ESC]キーで中止 |
| #MKDIR | ディレクトリの作成を行う。実行すると、窓を開いて入力を促す。設定した結果を決定する場合は [CR]キーを押す。[ESC]キーで中止 |
| #R-NAME | ファイル名およびディレクトリ名の変更を行う。実行すると、窓を開いて入力を促す。設定した結果を決定する場合は [CR]キーを押す。[ESC]キーで中止 |
| #CHDRIVE | ドライブ(A:～Z:)の変更。変更するときは、選択する窓が開くので、カーソルキーで選択するか、または[A]～[Z]を直接入力することにより行う。なお、ここではドライブのチェックを行っているので、フロッピーディスクが未挿入の場合等は変更しない |
| #HELP-INF | 画面の上方に、ファイル数またはファイル長等の表示を行う。.Disp-Levelの設定を変更する働きがある |
| #HELP-DRV | 画面の下方に、表示されているディスク容量の表示の有無を変更する |
| #TREE | 画面をツリー表示する。ツリー表示の状態では、#M-ALL(すべてのマーク)のみが実行できない。ツリー表示実行開始時にMALLOCでメモリの確保を行っているので完了するまでに時間がかかる。[ESC]キーで中止<br>また、ツリー表示を行うドライブは、ディレクトリ数＋ファイル数が4999以下であることが条件 |

この他にも、以下のような内部コマンドがあります。詳しくは、MF.X添付のドキュメントファイルをご覧ください。

| 内部コマンド名 | 内　容 |
|---|---|
| #M-ALL | すべてのマーキング |
| #M-ONE | マーキング。[SPACE]キーと同様の処理を行う |
| #M-ALL-DIR | (ディレクトリを含む)すべてのマーキング |
| #M-ONE-DIR | (ディレクトリを含む)マーキング |
| #RE-ATR | ファイル等の属性を変更する |
| #FIND-FN | ファイル名(ディレクトリ名)を検索する |
| #0EJECT<br>#1EJECT | X68000本体のフロッピーディスクドライブのフロッピーディスクの排出を行う |
| #EJECT | 拡張イジェクト。MOの排出を行う |
| #CD-Direct= | 指定したドライブの、指定したディレクトリに直接移動 |
| #CD-Parent | 1つ上の親ディレクトリに戻る |
| #RE-LOAD | ディレクトリ(FAT)の再読み込みを行う |

| | |
|---|---|
| #CLR-mark | マークしたものをすべて解除 |
| #CLR-disp | 画面を再表示する |
| #HELP-COM | 基本コマンドの登録キーの一覧を窓を開いて表示する |
| #HELP-INF | 画面の上方に、ファイル数またはファイル長等の表示を行う |
| #C-RIGHT | 著作権等の表示を行う |
| #0GVram | グラフィック画面を表示する |
| #1GVram | |
| #2GVram | |
| #3GVram | |
| #VD-UP | 表示画面の縮小を行う |
| #VD-DW | 表示画面の拡大を行う |
| #LS-UP | [↑]キーと同様の処理を行う |
| #LS-DW | [↓]キーと同様の処理を行う |
| #LS-LT | [←]キーと同様の処理を行う |
| #LS-RT | [→]キーと同様の処理を行う |
| #LS-RU | [ROLL UP]と同様の処理を行う |
| #LS-RD | [ROLL DOWN]と同様の処理を行う |
| #CR-ENTER | [CR]と同様の処理を行う |
| #RE-SET | Pa-0で設定したものを変更する |
| #END-MF | MF.Xを終了する |
| #WILD-DISP | 表示するファイル名を、ワイルドカード等で限定する |
| #SORT-quic1 | ダイレクトソート |
| #SORT-quic2 | |
| #SORT-direct | |
| #KEYboad-LOCK | キーボードロック |

# LZX.X

## .X、.Rファイルを実行可能なまま圧縮するツール

【概要】……………………

パソコン通信を実際にしたことがない人でも、最近の『Oh! X』誌などでLH.XやLHA.Xの話は聞いたことがあるかもしれません。これらのツールを使えば、たくさんのファイルを圧縮して格納することで、フロッピーディスクの枚数を減らすことができます。また、パソコン通信では、ファイルサイズが小さくなれば必然的に通信時間を減らすことができ、LH.X、LHA.Xは大活躍です。

しかし、LH.X、LHA.Xで圧縮されたファイルは確かに驚異的に小さくなりますが（どのくらい圧縮されるかはLHAのページを参照してください）、そのまま実行することはできません。ファイルの中身を見るなら、SEE等のLZHファイルの自動展開機能付きビューワがありますが、実行ファイルについては、いったんフロッピーディスクなり、ハードディスクに展開してからでなければ使用できません。

そこで、実行ファイルを圧縮した状態のまま実行できないかという発想が生まれます。これによって限りあるディスクの磁性面を節約しようというもので、MS-DOS上で動作する“LZEXE.EXE”が、このツールの原点となっています。

圧縮された実行ファイルには、専用の展開ルーチンが付加されてディスクに書き込まれており、この展開ルーチンから実行が開始されます。展開ルーチンにより、圧縮されている部分が展開され、元のプログラムがメモリ上に復元されたあと、本来の実行開始番地から実行が開始されます。

このLZEXE.EXEの仕組みおよび圧縮方式を参考に、Human68kでも同様のことを実現したプログラムがLZX.Xです。LZX.Xの圧縮率はものによって違いますが、20%から60%程度まで圧縮してくれます。また、展開する際は大きなファイルでも数秒で完了しますから、通常使用する場合はほとんど待たされることはありません。フロッピーディスクなどのファイルの読み込みが遅い媒体上ではファイルサイズが小さくなったため、かえって起動が速くなる場合もあります。

補足：
LZX.X 以外にも、同様の発想で「特定のデータを圧縮するもの」「汎用のデータ圧縮をしてくれるもの」「テキストデータ専用のもの」などが存在しています。市販ソフトウェアでもこの圧縮技術を使い、ディスクの節約をして、ユーザをディスク交換の煩雑さから救おうとしています。最近「TAKERU」で発売された「サークⅡ」はLZX.Xを使用していました。

【作者】…………………… F&Iさん　NIFTY-Serve　QFH00464

**【推薦します】**…………　「HDDの容量が足りないよー」という人、フロッピーベースで使っているので「システムディスクが1枚を超えてしまう～」という人、これで実行ファイルをボコボコに叩きのめしましょう(ぉぃ)。ストレス解消にこの1本(違うって)。とにかく使って損はないツールです。ちなみにフロッピーで使っている人は総合的に起動がちょっとだけ速くなる！

MAX BBS　てぃん

**【使用する前に】**………　本書添付ディスクのLZXには、LZX.XとLZXH.Xの2種類があります。LZX.Xは、シャープ純正のCOMMAND.Xでの使用を想定して、LZX自身が引数に与えられたファイル名のワイルドカードを展開します。以後の説明では、こちらを使うことを前提にして説明を進めます。また、LZXH.XはHUPAIR（補足参照）準拠版です。ワイルドカード展開を行うシェルで使用することを想定していますので、LZXH.X自体はワイルドカードの展開機能を持ちません。その分、ファイルサイズが小さくなっています。

補足：HUPAIR
　Human68k User Program Argument Interface Regulationsの略称で、板垣史彦氏により提唱された、Human68k上でのプログラム間の引数受け渡し方法に関する規定。LZXH.Xは、この規定に則って引数の受け渡しをします。

**【主な使用法】**…………　LZX［ファイル名］

　引数で指定された実行ファイルを圧縮します。
　LZXで圧縮できるのは、次のファイルです。

（1）Human68kで動作する実行ファイル[注1]
（2）Human68kのデバイスドライバ[注2]
（3）SX-WINDOWで動作するモジュールタイプOBJC/O/R[注3]形式の実行ファイル

　なお、圧縮しようとすると逆にサイズが大きくなってしまう場合や、圧縮できてもディスク上のセクタサイズが変わらない場合には、次のようなメッセージが表示され、圧縮ファイルは作られません。

**セクタ使用数の改善ができませんでした。**

　Human68k標準のコマンドでは、DUMP.X、SYS.Xなどの場合、こうなります。
また、すでにLZXで圧縮されている場合は、

**既にLZX.Xで圧縮されています.**

と表示され、二重の圧縮はできません。

注：
　1）実行ファイルでもバインドされたものについては、そのままでは圧縮できない。「使用法　2. バインドされたファイルを圧縮する場合」を参照のこと。
　2）圧縮したデバイスドライバを使用するには、同封のLZXLOADER.SYSを併用する。なお、HUMAN.SYSの圧縮はできない。
　3）“HDﾌｫｰﾏｯﾄ.x”はOBJXタイプなので、圧縮できない。

**【書式】**…………………　lzx［オプションスイッチ］［ファイル名］

　オプションスイッチが指定されていない場合は、ファイルの圧縮を行います。ファイル名には、複数のファイルを指定したり、ワイルドカードを利用したファイル名の指定も可能です。この場合は、各々のファイルについて、LZXの処理を実行します。

**【オプションスイッチ】**

| オプションスイッチ | 内　容 |
|---|---|
| -d | 圧縮された実行ファイルを展開して、元のファイルにする |
| -r | “.x”形式の実行ファイルを圧縮して作るファイルを“.r”形式の実行ファイルにする |
| -l | ファイルの情報を表示する。LZXで圧縮されているかどうかや、元のファイルのサイズ、圧縮率などを知ることができる |
| -x数値 | 圧縮ファイルの実行時に、LZXの展開ルーチンがメモリ不足等の理由で本来のプログラムを起動できない場合、デフォルトでは32767をEXITコードに返すようになっているが、数値で指定した番号を返すようにする(補足参照) |

補足：
　LZX展開ルーチンがメモリ不足で本来のプログラムを起動できなかった場合、

　　　　メモリ不足の為、起動できません.

と表示され、通常は32767のEXITコードが返される。
　本来のプログラムでこのEXITコードと同じ番号を使っていた場合でも、画面を見ていれば、本来のプログラムの実行中にエラー等で終了したのか、プログラムが起動する前にLZX展開ルーチンでエラーになって終了したのかを知ることができる。
　しかし、バッチファイルなどで自動実行するような場合には、EXITコードでエラーを区別できなくてはならない。このため、“-x”オプションにより、本来のプログラムの返すEXITコードとぶつからない番号に変える必要がある。

**【使用法】**…………………　LZXの使用は、ファイルを圧縮する場合がほとんどですが、Bup.x（134ページ参照）などを使って行われる実行ファイルのアップデートでは、元の実行ファイルが必要（補足参照）になるので、“-d”オプションを使って圧縮ファイルを展開する必要もあります。
　ここでは、通常の実行ファイルの圧縮、バインドされたファイルの圧縮、圧縮されたファイルの展開について説明します。
　なお、言うまでもないことですが、オリジナルのシステムディスクで、実行ファイルの

圧縮や展開などの作業を実施するのは避けてください。

補足：
　プログラムのバグ修正やバージョンアップにおいて、わずかな修正なら、新しいものは、元のファイルと大部分が同じで、一部のみ違うものになり得る。ところが、これらをLZXで圧縮してしまうと、ファイル全体にわたって大きく違ったものとなってしまう。このため、Bdif/Bupによるアップデートでは、LZXで圧縮する前の元ファイルで実行する必要がある。

## 1. 実行ファイルを圧縮する場合

　Human68kのコマンド類など、実行ファイルを一括して圧縮する場合、実行ファイルがあるディレクトリをカレントディレクトリにして、

LZX　＊.X　＊.R[CR]

とします。圧縮されたファイルは、元のファイルと同じファイル名で書き込まれます。元のファイルは、拡張子部分が以下のようにリネームされて残されます。

| 元のファイル名 | | 圧縮後の元のファイル名 |
|---|---|---|
| foo.x | → | foo.ox |
| foo.r | → | foo.or |
| foo.sys | → | foo.os |

例）
C:¥binディレクトリ内の実行ファイルをすべて圧縮する場合

C:[CR]　　←カレントドライブをドライブ“C:”に移動
CD ¥bin[CR]　　←カレントディレクトリを“¥bin”に移動
LZX　＊.X　＊.R[CR]　←実行ファイルを圧縮

　これにより、たとえば“ED.X”という実行ファイルは圧縮されて“ED.X”に、元のファイルは“ED.OX”となります。
　Human68kの標準コマンドが入っているディレクトリで行った場合、FORMAT.Xのようなバインドされたファイルや、SYS.Xなどのセクタサイズが減らないファイルなど、LZX圧縮がかからないものもありますが、それでもかなりのファイルが圧縮されるでしょう。
　LZX実行後は、念のため、圧縮された実行ファイルを起動して、正しく動作するかどう

か確認してみてください。万が一、正常に動作しない圧縮ファイルがあったら、その圧縮ファイルを削除して、“.ox”や“.or”の拡張子で残っている元のファイルを“.x”や“.r”にリネームすればよいのです。不具合がないようなら、“.ox”や“.or”の拡張子にリネームされた元のファイルは削除してかまいません。

DEL　＊.ox[CR]　←x形式の元ファイルを削除
DEL　＊.or[CR]　←r形式の元ファイルを削除

Human68k標準の実行ファイルだけでも、ディスクスペースはかなり節約できるでしょう。

カレントディレクトリとは別のディレクトリにあるファイルでも、そのファイルのパス名を指定してLZXを実行することができます。この場合、圧縮されたファイルはカレントディレクトリに作られますが、別のディレクトリにある元のファイルはリネームされません。十分な容量のRAMディスクがあれば、こちらの方法のほうが処理が速いでしょう。

例)
カレントディレクトリが“G:¥”で、別のディレクトリ“C:¥bin”にある実行ファイルを圧縮する場合

LZX C:¥bin¥＊.x C:¥bin¥＊.r[CR]

圧縮されたファイルは、カレントディレクトリ“G:¥”に元のファイルと同じファイル名で書き込まれます。“C:¥bin”にある元のファイルはリネームされません。圧縮された実行ファイルの動作を確認したら、元のファイルに上書きしてかまいません。

COPY　＊.x C:¥bin[CR]　←元のファイルに上書きコピー
COPY　＊.r C:¥bin[CR]　←元のファイルに上書きコピー

## 2. バインドされたファイルを圧縮する場合

COMMAND.XやFORMAT.Xは、複数の実行ファイルからできており、バインド(BIND)という機能を使って1つのファイルにまとめられています。このようなファイルをLZXで圧縮しようとした場合、

**このファイルはBINDされている可能性があります。**

と表示され、圧縮されません。

この場合、Human68k付属のコマンド"BIND.X"を使って、いったんバインドされたファイルを個々の実行ファイルに分解し、各々にLZXをかけて圧縮したあと、再びBIND.Xコマンドによりバインドし直すという手順を踏まなければなりません。

例)

FORMAT.Xを圧縮する

FORMAT.X（ver2.11を前提にしています）では、FORMAT.X、FORMAT1.X、FORMAT2.X、FMTDPB.FMT、FMTHD.FMTの5つのファイルがバインドされています。BIND.Xコマンドにより、FORMAT.Xからこれらを取り出します。なお、元のFORMAT.Xをカレントディレクトリに置いて作業をすると、バインドされているファイルの中にあるFORMAT.Xが同じファイル名の元のFORMAT.Xを破壊してしまうので、元のFORMAT.Xをリネームしておくか、次のように別のディレクトリ上で作業します。

元のFORMAT.Xが"C:¥bin"にあり、カレントディレクトリが"G:¥"であるとして、

```
G:/>BIND /X C:¥bin¥FORMAT.X FORMAT.X FORMAT1.X FORMAT2.X
FMTDPB.FMT FMTHD.FMT[CR]
```

として各々のファイルを取り出します。

このうち、FORMAT.X、FORMAT1.X、FORMAT2.Xの3つの実行ファイルに対してLZXをかけます(実際には、FORMAT.Xはサイズが改善されないので圧縮されませんが)。

```
G:/>LZX FORMAT.X FORMAT1.X FORMAT2.X[CR]
```

あとはバインドし直すだけです。バインドするファイルにFORMAT.Xがあるので、新しいファイル名はformat_new.xなどとして、次のように実行します。

```
G:/>BIND /O format_new.x FORMAT.X FORMAT1.X FORMAT2.X FMTDPB.
FMT FMTHD.FMT[CR]
```

新しいformat_new.xの動作に問題になければ、元のFORMAT.Xの名前で上書きしてもいいでしょう。

```
G:/>COPY format_new.x C:¥bin¥FORMAT.X[CR]
```

なお、フリーソフトで“LZXBIND.X”という、バインドファイル圧縮用のサポートツールを使えば、もっと簡単にBINDファイルの圧縮ができるようです。

COLUMN

**FORMAT.Xのバインド**

FORMAT.XをBINDコマンドの“/L”オプションスイッチで表示させても、“- system reserved file-”となってファイル名は見えないようになっています。これは、重要なコマンドなので、ユーザに触って欲しくないということなのでしょう。ここでは、最初から5つのファイル名を知っているとして話を進めてしまいましたが、実はディスクエディタでFORMAT.Xの中身を見てファイル名を探り当てたのであって、正規の手段ではありません。バインドされたファイルの例としては、あまり適当ではないのですが、FORMAT.Xはサイズが90Kバイトほどあり、LZXにより半分近くに圧縮できて効果が大きいということで取り上げました。

バージョンによっては(もしかしたら同じバージョン表示のものでも)、うまくいかないかもしれません。参考程度と思ってください。

### 3．圧縮した実行ファイルを展開する場合

LZX -d［圧縮されたファイル名］

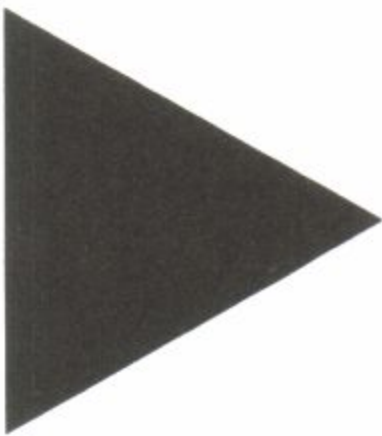
圧縮された実行ファイルを展開し、元のファイルをカレントディレクトリに作成します[注]。このとき、展開元となる圧縮ファイルがカレントディレクトリにある場合はリネームされます。圧縮ファイルのリネームは、ファイルの拡張子により以下のようになります。

| 圧縮ファイル名 | 展開後の圧縮ファイル名 |
|---|---|
| foo.x | foo.lzx |
| foo.r | foo.lzr |
| foo.sys | foo.lzs |

注：元の実行ファイルにデバッグ用のシンボル情報がついていても、LZXで圧縮する時点で削除されてしまうので、展開時にはシンボル情報を復元することができない。このため、元のファイルと完全に同じものにならない場合がある。

例）カレントディレクトリにある圧縮された実行ファイル“ED.X”を展開する場合

LZX -d ED.X[CR]

展開されたファイルは“ED.X”になり、圧縮ファイルは“ED.LZX”となります。

# SEE.X

## LZH(LHA)ファイルの内容も確認できるファイルビューワ

**【概要】**

SEEの特徴は、Pht.1のように、ファイル選択時に画面上部にファイル一覧が表示されるとともに、まだファイルを選択していない状態でも、カーソルが位置しているファイルの先頭の内容が画面下半分の領域に表示されるという点です。カーソルを動かせば、各ファイルの内容が次々に表示されるので、目的のファイルを大変容易に探すことができます。

また、目的のファイルがLZH形式の圧縮ファイルの場合も、SEE内部で自動展開して表示してくれるので、ファイルの内容を確認するために、いちいちLHAを呼び出して展開したあと、ビューワを起動するといった手間がかかりません。

このファイルビューワ機能の他にも選択したファイルの拡張子によって指定のプログラムを起動させたり、特定のキーにプログラムを登録して簡単なキー操作でプログラムを起動するなどの簡易ファイラとしての機能も備えています。SEEの機能の特筆すべきは、このプログラムの起動を行うとき、選択したファイルがLZH形式の圧縮ファイル内だった場合に、RAMディスクなど、あらかじめ指定しておいたワークディレクトリ上に展開するように指定できることです。この機能により、LZH形式の圧縮ファイルになっている文章のエディットや、PICファイル等の画像ファイルを表示させたりする場合でも、いちいちLHAを呼び出して展開する手間がかかりません。

**Pht.1**　SEEをオプションスイッチなしで起動したときの画面

```
SEE : Glancing at files v0.31+1 Copyright 1991 HARPOON

           HUMAN.SYS    COMMAND.X     AUTOEXEC.BAT  AUTOEXEC.BAK  CONFIG.SYS
wind.SNG   septem.SNG   BIN           ATTRIB.X      CHKDSK.X      COPY2.X
CUSTOM.X   DISKCOPY.X   MF.DEF        MF.X          MF_BIN.DEF    ish121.Lzh
BdifSet.Lzh Fu204e2_.Lzh FUTYPE.LZH   MF_X_205.Lzh  lzx04341.lzh  see031_1.lzh
DC2_065.LZH 118SUPER.LZH tsort255.Lzh dedit.doc     dedit.x       q2
see.x      see.def      dc.r          DEDIT.coo

C:¥                           [Sort=N/Disp=C/LZH=A/Tab=8]  (  1/ 34 )
see.doc                       A----- 91/07/29 00:31:00   15142 bytes

◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆↓
      SEE  : Glancing at files  [ v0.31 ]  91/07/28↓
◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆↓

     SEE とはファイルの頭の部分を、ちょろちょろ覗きながら、ファイルを選択し↓
     各種処理を行なうものです。↓

     LZH ファイルの自動解凍表示・解凍が可能です。↓
          [圧縮フォーマット : -lh0- -lh1- -lh4- -lh5-] ↓
          [ヘッダレベル     : 0-2                  ] ↓

     さらに、アーカイブ内のファイルを圧縮したままで、エディットしたりPICファイルを↓
```

**【作者】**……………………HARPOON（長沢聡一郎）

**【推薦します】**…………　パソコン通信などをやっていると、圧縮されたファイルがどんどん溜まっていきます。解凍しないと中に何が入っているのかわからず、いざ必要になったときに非常に苦労していました。しかし、SEEを使えば、圧縮されたままの状態で中のドキュメントを読んだり、PIC画像を表示させたりできます！　管理が非常に楽になりました。

MAX BBS　Zeno

私は不精なので、プログラムを作るときなど、test1.c、test2.cのように適当なファイル名をつけてしまうので、あとでファイルを探すとき、名前だけでは頼りにならず、いつもSEEのお世話になっています。

MAX BBS　BEEPs

**【使用する前に】**………　SEEには、各種動作を指定する定義ファイルsee.defが用意されています。ビューワとして使用するなら、とりあえず添付ディスクのsee031_1.lzhに含まれているsee.defのままでよいのですが、ファイラとしての機能を使う場合は、LZH形式の圧縮ファイルを自動展開するときにワークとして使用するディレクトリを各自の環境にあわせて設定する必要があります。

see.defの中ほどに、

```
*==== LZH自動展開用ワークディレクトリ ====
#TEMP   K:¥
*==== カラーパレット =====================
```

という行がありますので、ED.X等のエディタを使って“K:¥”の部分をRAMディスクなど、できるだけ高速のディスクに変更してください。

また、ディレクトリを含めて指定する場合には、

```
#TEMP   D:¥TMP¥
              ↑
```

のようにディレクトリ名のあとに、必ず“¥”をつけておきます。

この他にも、see.defによりSEEの機能をカスタマイズすることができますが、とりあえず、このままでよいでしょう。カスタマイズについては、「使用法　4.　カスタマイズのしかた」で紹介します。

SEEは、起動時にsee.x自身のあるディレクトリと同じディレクトリからsee.defを読み込むので、see.xをディレクトリbin等の下にコピーした場合は、see.defも同じディレクトリにコピーしておかなければなりません。また、see.xをタイプ量を減らす目的でs.x

のような名前に変更した場合は、see.defもs.defというように変更しておく必要があります。

**【主な使用法】**…………　SEEを起動する場合は、

SEE[CR]

とします。すると、ファイル一覧が表示され、その中からファイルを選択するモードでSEEが起動します。ビューワやファイラ機能は、ここからファイルを選択して起動します。

**【書式】**……………………

| | | |
|---|---|---|
| SEE | | ← 何も指定しない |
| SEE | ディレクトリ名 | ← ディレクトリを指定 |
| SEE | ファイル名 | ← ファイル名を指定 |

**【使用法】**………………

## 1．ファイル一覧からファイルの内容を見る場合

SEE[CR]　または、　SEE ディレクトリ名[CR]

ディレクトリ名を指定しない場合はカレントディレクトリを、ディレクトリ名を指定した場合はそのディレクトリ配下のファイル一覧が画面上半分に表示されます。SEEでは、これを「ファイル選択フェーズ」と呼んでいます。画面下半分は、ファイル一覧のうち、カーソルが現在あるファイルの内容が表示されています。カーソルを移動して目的のファイル名上で[CR]キーを押すことでファイルの選択を行います。

カーソルの位置しているファイルがディレクトリやLZH形式の圧縮ファイルの場合は、サブディレクトリのファイル名や圧縮されているファイル名が表示されます。ファイルを選択すると、サブディレクトリに移動したり、LZH形式のファイルの自動展開が行われるようになっており、LZH形式の圧縮ファイルをディレクトリの一種と見れば、直感的に操作できるでしょう。

表示内容と操作についてまとめると、以下のようになります。

**●通常のファイルの場合**

ファイル選択フェーズ中の表示

ほとんどの場合がこのファイルですが、ファイルの先頭から十数行がそのまま表示されています。[M]キーを押すことにより16進数によるダンプ表示に切り替えることができま

すので、ファイルがテキストファイルでなかった場合でも、ある程度の内容が確認できます。

なお、そのファイルを選択して内蔵ビューワを起動すればファイル全体を見ることができます。内蔵ビューワの起動で待たされることはほとんどありません。

ファイル選択時の動作

[CR]キーを押すと「ビューワフェーズ」になります。ビューワの使用方法については、「使用法　2.　ビューワ」で説明します。

[SPACE]キーを押すとエディタが起動されます。この他にもファイル選択のキーにより、いろいろな機能を起動することができます。これらは、まとめて「使用法　3.　簡易ファイラ」で説明します。

**●ディレクトリの場合**

ファイル選択フェーズ中の表示

サブディレクトリ内のファイル名が表示されます。

**Pht.2**　ファイル一覧でディレクトリを選択したときの画面。画面下半分には、ファイル内容のかわりにサブディレクトリ内のファイル名が表示される

```
SEE :.Glancing at files v0.31+1 Copyright 1991 HARPOON

see.doc      HUMAN.SYS     COMMAND.X     AUTOEXEC.BAT  AUTOEXEC.BAK  CONFIG.SYS
wind.SNG     septem.SNG    BIN           ATTRIB.X      CHKDSK.X      COPY2.X
CUSTOM.X     DISKCOPY.X    MF.DEF        MF.X          MF_BIN.DEF    ish121.Lzh
EditSet.Lzh  FU204e2_.Lzh  FUTYPE.LZH    MF_X_205.Lzh  lzx04341.lzh  see051_1.lzh
DC2_065.LZH  118SUPER.LZH  tsort255.Lzh  dedit.doc     dedit.x       q2
see.x        see.def       dc.r          DEDIT.coo

C:¥                              [Sort=N/Disp=C/LZH=A/Tab=8]  (  9/ 34 )
BIN                              -D----  93/02/12 23:50:56 <dir>

.            ..            ATTRIB.X      CHKDSK.X      COPY2.X       CUSTOM.X
DISKCOPY.X   DUMP.X        FC.X          FIND.X        FORMAT.X      KEY.X
MORE.X       USKCGM.X      ED.X          DICM.X        DRIVE.X       WHERE.X
MOVE.X       BACKUP.X      ED.HLP        sel.x         ish.x         ish&zoo.doc
COPYALL.X    x68k.x        banner.x      GVON.X        gram.x        verf.x
view.rc      view.x        view.doc      COMMAND.X     MF.DEF        MF_MAN.LZH
bdif.x       MF.X          MF_BIN.DEF    FU.CNF        FU.Env        bup.x
Fu.x         Ecv.x         Fu.Man        Ecv.Man       Fu_Cnf.Man    bfd2rx.x
FU.BAK       lzx.x         lzxh.x        see.doc       see.x         dc.r
dedit.doc    dedit.x       dirsort.mcr   DCACHE2.DOC   DCACHE2.R     BASIC.X
basic.BAK    AUDIO.FNC     GRAPH.FNC     MOUSE.FNC     MUSIC.FNC     IMAGE.FNC
basic.cnf    nocross.bas   cross1.bas    cross2.bas    cross3.bas    cross4.bas
cross5.bas   mojitabl      a_maze.x      bo.x          MENU.BAK      0.BAT
1.BAK        2.BAT         3.BAT         4.BAT         5.BAT         6.BAT
1.bat        2.BAK         3.BAK         MENU.BAT      INVADER.X     INVADER.DOC
NIGHTS.ISH   NIGHTS.Lzh    NIGHTS.X      NIGHTS.DOC    FIRE.PCM      KIN2.PCM
```

ファイル選択時の動作

[CR]キーを押すとサブディレクトリに移動し、再びファイル選択フェーズとなります。[SPACE][TAB]などのキーにより、いろいろな機能を起動することができます。これらは、まとめて「使用法 3.　簡易ファイラ」で説明します。

### ●LZH形式の圧縮ファイルの場合

圧縮されているファイル一覧が表示されます。

#### ファイル選択フェーズ中の表示

圧縮されているファイルの一覧が表示されます。表示内容はディレクトリとほぼ同じです。

#### ファイル選択時の動作

[CR]キーを押すと「LZHアーカイブファイル選択フェーズ」になります。画面上半分に圧縮されているファイルの一覧が表示されます。LZH形式圧縮ファイル内でファイルを選択するフェーズです。

この他、ファイル一覧では、以下のキーが使用できます。

| キー | 内容 |
|---|---|
| [D] | ドライブを切り替える<br>他のドライブ上のファイルを見たいときに使う。ドライブ名が表示されるので、カーソルで変更したいドライブを選択する |
| [ROLL UP]<br>[ROLL DOWN] | ファイル一覧のページを切り替える<br>ファイル一覧では、一度に48個までのファイル名が表示されるが、ファイルがこれより多い場合は、このキーで画面を切り替えることができる。ファイルがいくつあるかは、ファイル総数表示画面で確認できる（Fig.1） |
| [SHIFT]＋カーソルキー | カーソルを高速移動させる<br>ファイル一覧の画面上でカーソルを移動させると、カーソルが移動するたびに、ファイルの内容を表示するため、フロッピーディスクなどファイル読み込みが遅い装置のうえではカーソル移動が遅くなってしまう。このような場合、[SHIFT]キーを押しながらカーソル移動を行うと、ファイル読み込みがスキップされるので、カーソル移動が軽快になる。[SHIFT]キーを押している間は、Fig.2のように枠型のカーソルが移動し、[SHIFT]キーを離した時点で、本来のカーソルが移動してファイルの内容表示が行われる |
| [P] | 親のディレクトリに移動する |
| [S] | ファイル一覧で、ファイル名でソートするか、拡張子でソートするか、ソートしないかを切り替える。デフォルトではソートしない |
| [L] | LZH形式の圧縮ファイルを自動展開するか、しないかを切り替える。デフォルトでは展開する。展開しないようにした場合、200ページPht.3のようにLHA形式の圧縮ファイルの内容そのものが表示される |
| [Q] | see.xを終了する<br>現在ファイル一覧で表示しているディレクトリをカレントドライブ、カレントディレクトリにして終了する |
| [ESC] | see.xを終了する<br>ただし[Q]と異なり、いきなり終了せず、確認を求める<br>Y：起動ディレクトリで終了<br>N：終了しない<br>Q：今見ていたディレクトリで終了<br>の中から選択する<br>なお、LZHアーカイブファイル選択フェーズの場合は、通常のファイル選択フェーズに戻る |

**Fig.1** ファイル総数の表示

前のページがないことを示す
あれば“<”になる
次のページがあることを示す。なければ“)”になる
( 2 / 1 2 6 >
ファイル総数
カーソルのあるファイルの順番

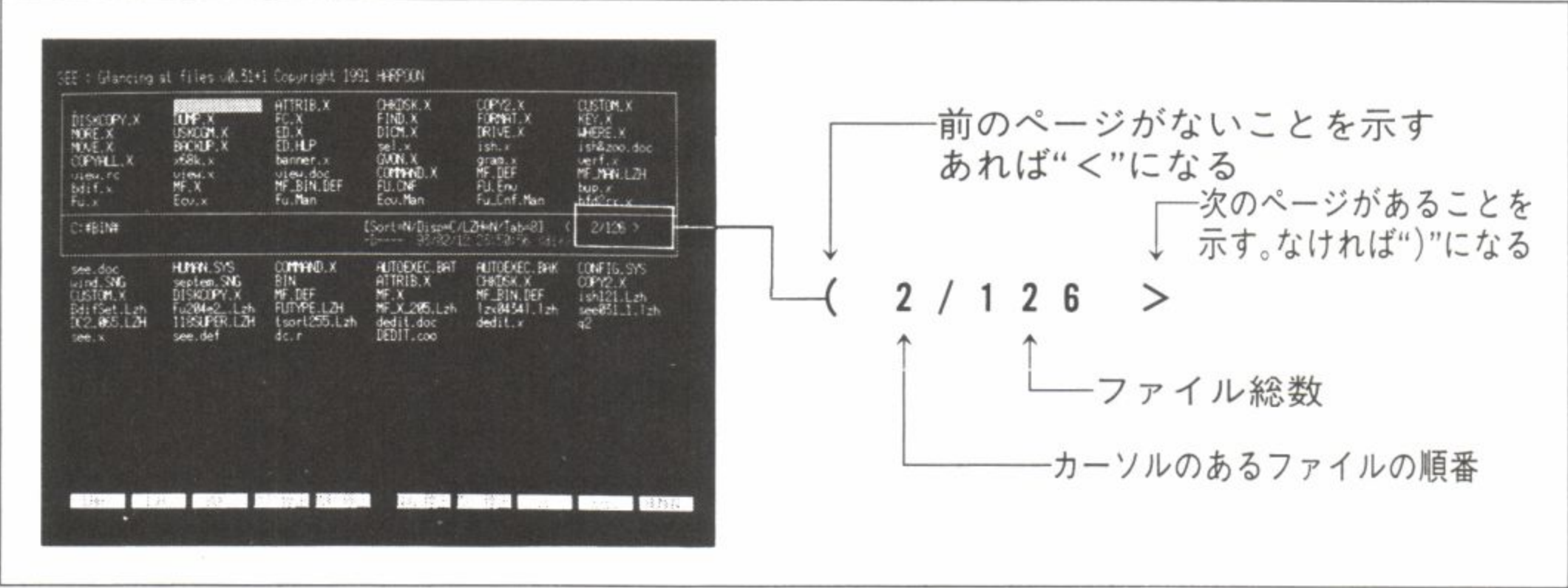



**Fig.2** [SHIFT]+カーソルキーによる移動時のカーソルの動き

| | | | | |
|---|---|---|---|---|
| . | .. | abc.def | def.ghi | dokode.def |
| z1.def | z2.def | bee.c | nanda | iraira.suru |
| kajiki.ddd | urusai | ninniku | tabetai | tekito.c |
| naitebaka | damatte | chikako | ijiwaru | |

[Sort=N/Disp=C/LZH=A/Tab=8] ( 3/211 >

```
#include <stdio.h
main(argc,argv)
        int argc;
```

そのファイルの内容
本当のカーソル
[SHIFT]キーを押しながら動かした仮カーソル

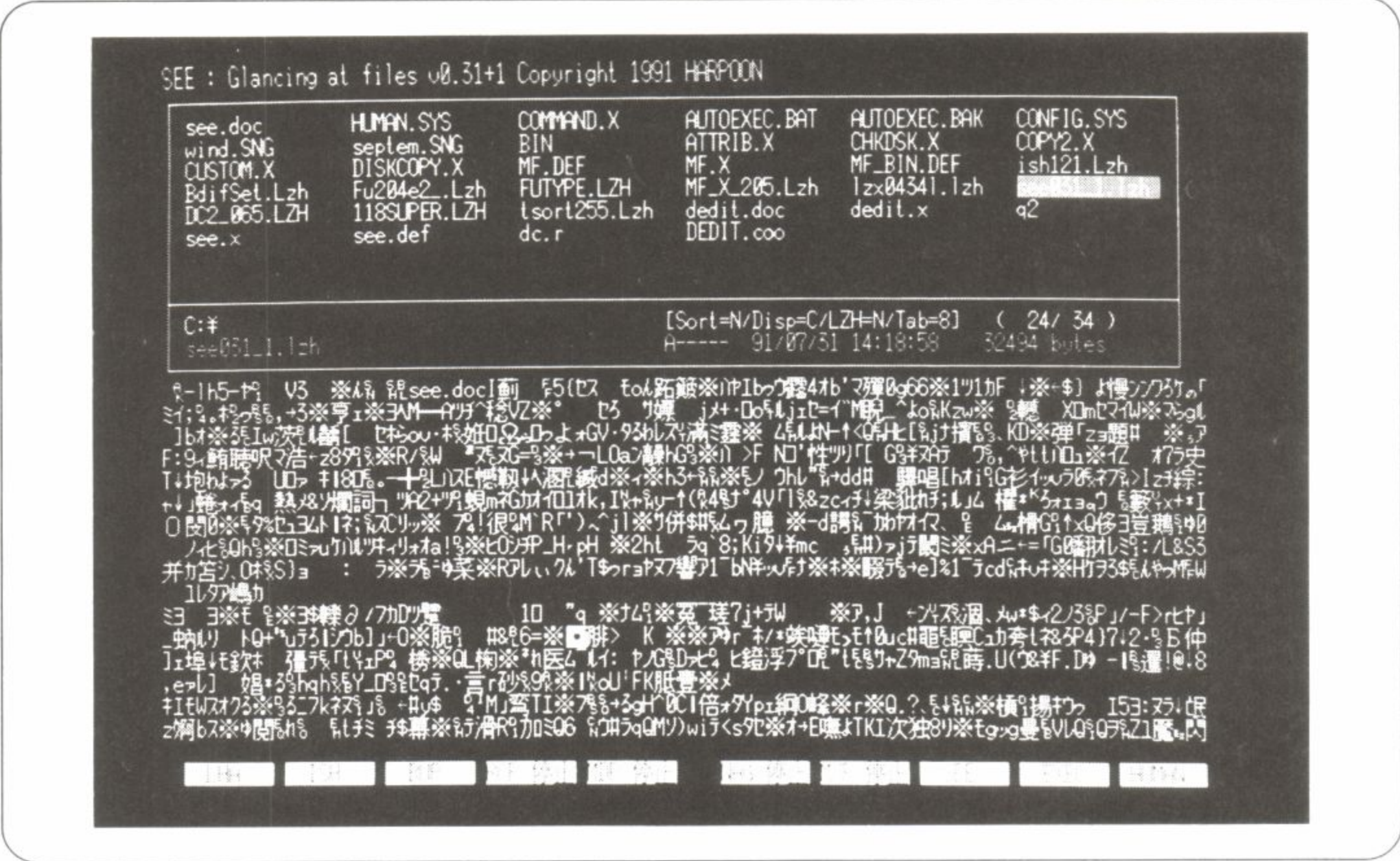
[SHIFT]キーを離すと

| | | | | |
|---|---|---|---|---|
| zl.def | z2.def | bee.c | nanda | iraira.suru |
| kajiki.ddd | urusai | ninniku | tabetai | tekito.c |
| naitebaka | damatte | chikako | ijiwaru | |

[Sort=N/Disp=C/LZH=A/Tab=8] ( 3/211 >

```
#include <oslib.h
```

ファイル内容も更新される
本当のカーソルが飛んでくる

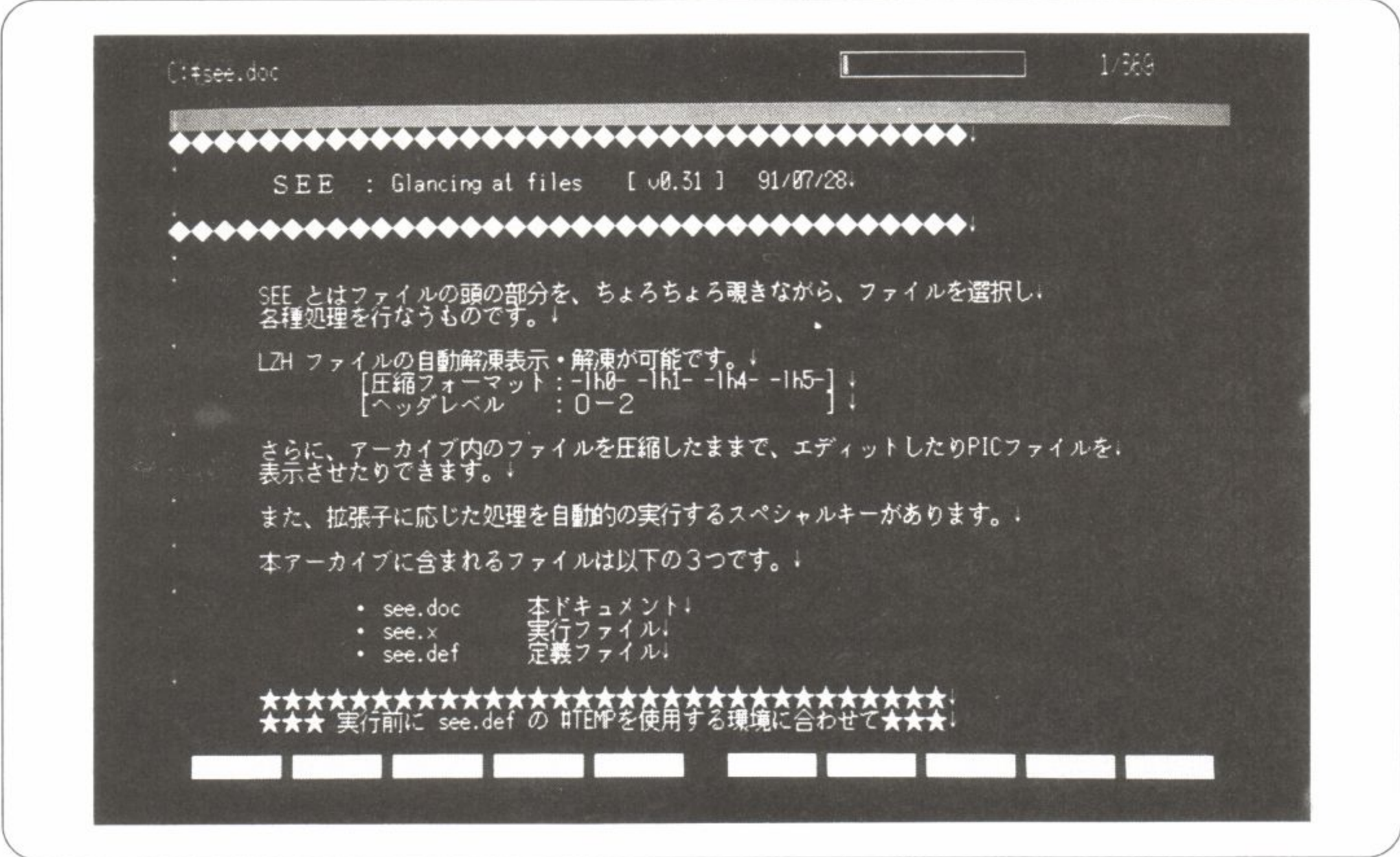
**Pht.3**　LHA形式の圧縮ファイルを自動展開しない場合の表示

## 2. ビューワ

SEE　ファイル名[CR]

ファイル名を指定してSEEを起動するか、ファイル一覧からファイルを選択してビューワが起動された場合、Pht.4のようにファイルの内容を見るためのビューワフェーズとなります。

SEEのビューワには検索やマーク、ジャンプ等の機能はありませんが、スクロールが速く、中身をちょっと見るには十分です。目的のファイルだと確認したら、ファイル選択フェーズに戻らなくても、そのまま[SPACE]キーを押してエディタを起動するといったことが行えます。また、見ているファイルがLZH形式の圧縮ファイルの中にあるファイルだった場合には、ワークディレクトリに展開してからエディタが起動されます。

Pht.4　ビューワ画面

```
C:¥see.doc                                         1/369

  SEE  : Glancing at files    [ v0.31 ]   91/07/28

SEE とはファイルの頭の部分を、ちょろちょろ覗きながら、ファイルを選択し
各種処理を行なうものです。

LZH ファイルの自動解凍表示・解凍が可能です。
        [圧縮フォーマット：-lh0- -lh1- -lh4- -lh5-]
        [ヘッダレベル    ：0−2                  ]

さらに、アーカイブ内のファイルを圧縮したままで、エディットしたりPICファイルを
表示させたりできます。

また、拡張子に応じた処理を自動的の実行するスペシャルキーがあります。

本アーカイブに含まれるファイルは以下の3つです。

      ・see.doc     本ドキュメント
      ・see.x       実行ファイル
      ・see.def     定義ファイル

★★★ 実行前に see.def の #TEMPを使用する環境に合わせて★★★
```

ビューワでは、以下のキーが使用できます。

| キー | 内　　容 | デフォルト |
|---|---|---|
| [↑][↓] | 上下のカーソルキーで高速スクロール | |
| [←][→] | 左右のカーソルキーで低速スクロール | |
| [ROLL UP]<br>[ROLL DOWN] | ページの移動 | |
| [HOME] | ファイルの先頭へ移動 | |
| [DEL] | ファイルの最後尾へ移動 | |
| [SP]または<br>[SHIFT]+[CR] | エディタ起動 | |
| [UNDO] | 外部ビューワ起動 | |
| [F] | ディスクエディタ起動 | |
| [M] | 文字表示と16進数によるダンプ表示の切り替え | |
| [T] | タブ幅の4桁と8桁の切り替え | 8桁 |
| [C] | 制御コードを表示するかしないかの切り替え | 表示 |
| [R] | 改行とファイル終端（EOF）の表示をするか、しないかの切り替え | 表示 |
| [V] | 左右カーソル[←][→]キーでのスクロール速度の変更<br>中速になる | 低速 |
| [ESC]<br>[CR] | ビューワの終了。ファイル一覧に戻る | |

## 3．簡易ファイラ

SEEのファイラ機能は、ファイル選択フェーズ[注]から指定のキーを押すことでエディタを起動したり、あらかじめ登録しておいたプログラムを起動するものです。

注：LZHの圧縮ファイル内の一覧からファイルを選択した状態で、これらの機能を呼び出した場合は、see.defでワークディレクトリとして指定したディレクトリにSEEがファイルを展開してからエ

ディタなどのプログラムを起動するので、ビューワと同様、圧縮ファイルをいちいち展開する手間がいらない。また、エディタおよびディスクエディタを起動した場合は、更新後のファイルを元のLZH形式の圧縮ファイルに圧縮し直す処理も行われる。

したがって、説明の都合上、「使用法 1. ファイル一覧からファイルの内容を見る場合」と区別しましたが、SEEの動作上は、特に区別はありません。

see[CR]
または、 see ディレクトリ名[CR]

で起動し、ファイル選択フェーズになっている状態で、ファイル一覧から目的のファイルにカーソルをあわせ、[CR]キーのかわりに以下のキーを押します。

| [F1]-[F10]、[XF1]-[XF5] | プログラマブルキー |
|---|---|

see.defで、各キーごとに登録されたプログラムを起動します。
デフォルトでは、以下のプログラムが登録がされています。

| キー | 起動プログラム名 | 内容 |
|---|---|---|
| [F 1] | lha | 指定のLZHアーカイブファイルを展開する |
| [F 2] | ish | 指定のishファイルをデコードする |
| [F 3] | bup | 指定のバイナリ差分ファイルを実行する |
| [F 4] | rcp | rcp(rcp形式データ演奏ソフト)を実行 |
| [F 5] | mdp | mdp(mdd形式データ演奏ソフト)を実行 |
| [F 6] | np | np(nag形式データ演奏ソフト)を実行 |
| [F 7] | mxp | mxp(mdx形式データ演奏ソフト)を実行 |
| [F 8] | see | 指定のファイルを展開後、SEEの実行<br>(LZHで二重に圧縮されている場合に有効) |
| [F 9] | command | 指定のファイルでCOMMAND.X起動<br>(バッチファイル等に有効) |
| [F10] | command | COMMAND.Xを単独で起動 |
| [XF1] | command del *.bak | .bakの拡張子のファイルを削除 |
| [XF2] | process | processコマンド実行 |
| [XF3] | drive | driveコマンド実行 |
| [XF4] | make | makeコマンド実行 |
| [XF5] | lhv | 現在のディレクトリでlhvを実行 |

see.defを変更し、よく使うコマンドなどを登録することで、SEEをより使いやすくすることができます。変更方法については、「使用法 4. カスタマイズのしかた」で紹介します。

| [BS] | スペシャルキー |
|---|---|

選択したファイルの拡張子ごとに、see.defで拡張子に登録されたプログラムを起動します。

デフォルトでは、以下のプログラムが登録されています。

| 拡張子 | 起動プログラム名 | 内　容 |
|---|---|---|
| .pic | apicg | pic形式の画像ファイルの表示 |
| .mki | mag | maki形式の画像ファイルの表示 |
| .mag | mag | mag形式の画像ファイルの表示 |
| .rcp | rcp | rcp形式の音楽データの演奏 |
| .sng | stor | ミュージ君、ミュージ郎の音楽データのrcp形式への変換 |
| .mid | itor | 標準MIDIファイルのrcp形式への変換 |
| .mdf | lzm | lzmで圧縮された音楽データの演奏 |
| .mdi | mdp | MIDIDRV用の音楽データの演奏 |
| .mdn | np | .mdnnpnag形式の音楽データの演奏 |
| .mdx | mxp | .mdxmxpmdx形式の音楽データの演奏 |
| .ish | ish | ish形式のファイルのデコード |
| .bfd | bup | バイナリ差分ファイルのアップデート実行 |

| キー | 内　容 |
|---|---|
| [SHIFT]+[CR]または[SPACE] | ファイラおよびエディタ起動<br>カーソルがディレクトリ上にある場合は、see.defで指定したファイラが起動される。デフォルトでは、tf.xが起動される。カーソルが通常ファイルにある場合はsee.defで指定したエディタが起動される。デフォルトでは、EDが起動される |
| [UNDO] | 外部ビューワ起動<br>see.defで指定したビューワプログラムを起動する。デフォルトではview.xが登録されている |
| [TAB] | LZH形式の圧縮ファイルの展開・ファイラ起動<br>LZHアーカイブファイル選択フェーズの場合は、カレントディレクトリに対してファイルの展開を行う。通常のファイル選択フェーズの場合は、see.defで指定したファイラプログラムが起動される。デフォルトではtf.xが登録されている |
| [F] | ディスクエディタ起動<br>see.defで指定したディスクエディタを起動する。デフォルトではdedit.xが登録されている |

## 4. カスタマイズのしかた

see.defを変更することにより、SEEの機能を各自でさらに使いやすいようにカスタマイズできます。ここでは、see.defで指定する内容について説明します。付属のsee.defで、必要なところのみエディタ等で変更するとよいでしょう。

### ●登録プログラムの変更

SEEから起動されるプログラムの指定は、各行に次のように記述していきます。

{定義名} ["表示文字列"] [-オプションスイッチ] {プログラム名　パラメータ等}

{定義名}：SEEでカスタマイズできる機能の名前です。

| | |
|---|---|
| EDIT | [SHIFT]+[CR]、[SPACE]で起動されるエディタ名 |
| DEDIT | [F]で起動されるディスクエディタ名 |
| FILER | [SHIFT]+[CR]、[SPACE]もしくは[TAB]で起動されるファイラ名 |
| VIEWER | [UNDO]で起動される外部ビューワ名 |

| | |
|---|---|
| LZHCOMP | “-u”オプション（後述）で使用されるLZH形式の圧縮プログラム名 |
| [F 1]〜[F10] | ファンクションキー |
| [XF1]〜[XF5] | XFキー |
| .xxx | [BS]で起動される拡張子 |

注：現バージョンのSEEは、LZH形式の圧縮ファイルの展開機能は内蔵しているが、圧縮機能は持っていないので、LHA等を使用する必要がある。

["表示文字列"]：画面最下行のファンクションキーに表示される文字列を指定します。表示文字列が指定できるのは、[F 1]-[F10]のファンクションキーだけです。

[オプションスイッチ]：オプションは、登録プログラムを実行する前後でSEEが行う処理を指定します。以下の指定ができます。

| オプション | 内　容 |
|---|---|
| -f | 登録プログラムに、選択したファイル名を引き数として渡さない<br>ファイル一覧画面で“abc.d”というファイル上にカーソルを持っていき、エディタを起動した場合、“EDIT ED.X”とsee.defに指定されていれば、ED.X abc.dとしてエディタが起動されるが、“EDIT -f ED.X”とsee.defに指定されていれば、ED.Xが起動されるだけで、エディタにファイル名“abc.d”は渡されない |
| -p | 登録プログラムに、現在のディレクトリをパラメータとして渡す |
| -d | 現在カーソルがあるファイルがディレクトリだった場合は、プログラムを起動しない |
| -s | 画面を(0,32)に初期化、ファンクションキー表示なしでプログラムを起動する |
| -g | グラフィックを表示状態にしてプログラムを起動する |
| -n | 画面をクリアしないでプログラムを起動する |
| -c | カーソル非表示状態でプログラムを起動する |
| -k | 登録プログラム終了時にキー入力を待つ。picなどのグラフィック表示プログラムで指定することにより、SEEに戻ってすぐに画面がクリアされるのを防ぐことができる |
| -l | 選択したファイルがLZH形式の圧縮ファイル内の場合、ワークディレクトリに自動展開してからプログラムを起動する |
| -u | 上の“l”オプションと組み合わせることで、プログラム終了時に定義名“LZHCOMP”で指定したLZH形式の圧縮プログラムを使って圧縮し直される |

{プログラム名　パラメータ等}：起動するプログラム名やパラメータを最大8個まで書くことができます。

### ●外部プログラムの設定例

エディタをEDからSuperEDに変更する場合

see.defの“EDIT -lu ed”の行を、“EDIT -lu supered”に書き換えます。オプションスイッチとして、“-lu”が指定されていますが、これは、SEEへの動作指示で、次のような意味です。

・LZH形式の圧縮ファイルの場合、自動展開する(“-l”オプションスイッチ)
・エディタ終了後、LHAで圧縮し直す(“-u”オプションスイッチ)

[F 4]キーを押したとき、選択ファイルを削除するようにする場合

付属のsee.defでは、「F04　”RCP 停止” -fl rcp -e」となっています。これを「F04　”　削除　” -d command del 」に書き換えます。オプションスイッチとして、“-d”が指定されていますが、これは、ディレクトリに対しては実行しないようにするためです。

[F 5]キーを押したとき、選択ファイルを“G:¥TMP”にコピーするようにする場合

コピー先は別に“G:¥tmp”でなくてもどこでもいいのですが、コピー先を指定することは、残念ながらSEEではできません。

また、SEEでは、起動プログラムに与えるファイル名の位置が変えられませんので、以下のようなバッチファイルを用意しておきます。

```
copy %1 G:¥tmp
```

これを、たとえば、“copytmp.bat”というファイル名でパスの通ったディレクトリに入れておきます。そして、付属のsee.defで

```
F05  ''MDP 停止'' -fl mdp -e
```

となっているところを、

```
F05  '' コピー '' -l command copytmp
```

に書き換えます。

オプションスイッチとして、“-l”が指定されていますが、これは、LHAの圧縮ファイルの場合には選択ファイルを自動展開する指定です。

SEEのファイラ機能は、ビューワの延長のようなもので、カスタマイズにいろいろ制限があるため、スマートにはいきませんが、工夫次第で、それなりにできるという意味で紹介しました。また、複数のファイルを指定してコピーしたり、コピー先を自由に変えるといったことも、残念ながら、SEEからはできません。そういう場合には外部ファイラを起動すればよいでしょう。

拡張子.basのファイルだった場合、X-BASICを起動するようにする場合
　付属のsee.defで、

```
*==== スペシャルキー[BS]定義 =============
```

として拡張子定義をしてある行に、

```
.bas      -l basic
```

を追加します。

**●デフォルトの変更**

　ワークディレクトリの指定や画面の色、SEEの動作のデフォルト値などもsee.defで変更できます。

{定義名}{設定値}

| | |
|---|---|
| #TEMP | 自動展開のためのワークディレクトリ名の指定<br>RAMディスクなどの高速ディスクをドライブ名とともに指定する。ただし、最後に必ず“¥”をつけること |
| #COLOR0 | 背景色の指定 |
| #COLOR1 | ディレクトリおよびタイトル色 |
| #COLOR2 | LZH形式の圧縮ファイルおよび改行マークの色 |
| #COLOR3 | 通常ファイルおよびテキスト文字の色 |
| #COLOR4 | カーソルの色 |

注：これらの指定は、16進数4桁で指定する。ちなみに、0000が黒、FFFFが白となっている。

| | | | |
|---|---|---|---|
| @DISP | 表示モード | 0：文字表示 | 1：16進ダンプ表示 |
| @TAB | タブ文字数 | 0：8桁 | 1：4桁 |
| @SORT | ファイル名ソート | 0：ソートしない<br>1：ファイル名の順でソートする<br>2：拡張子の順でソートする | |
| @LZH | LZH自動展開 | 0：自動展開しない | 1：自動展開する |
| @CLEAR | 終了時の画面クリア | 0：クリアしない | 1：クリアする |
| @CREOF | 改行・EOFの表示 | 0：表示しない | 1：表示する |
| @CTL | 制御文字の表示 | 0：表示しない | 1：表示する |
| @SPEED | ビューワでの[←][→]キーのスクロール速度 | 0：低速 | 1：中速 |
| @ENDCK | 終了時[ESC]の確認<br>確認なしの場合は、SEE起動時のディレクトリに戻って終了する | 0：確認しないで終了 | 1：確認する |

**●デフォルトの設定例**

画面の色を、灰色をバックに文字を暗緑色で表示するように変更する場合

付属のsee.defで、

「#COLOR0 0000」としてある行を「#COLOR0 7BDE」に
「#COLOR3 FFFE」としてある行を「#COLOR3 2800」に

変更します。元の#COLOR指定は、行の頭に“＊”がついていてコメントになっていますが、変更後は、色の指定を有効にするために、“＊”を取っておきます。

**ファイル一覧を、ファイル名でソートして行うようにする場合**
付属のsee.defで“@SORT　0’としてある行を“@SORT　1”に書き換えます。

**【他のプログラム】……** 最後に、SEEは、いろいろな特徴を持っていますが、LZH形式の圧縮ファイルをそのまま見ることができるビューワという点で、もう１本紹介しておきましょう。

**＜プログラム名＞lhv**

| **＜作者＞けんぢ** | | | |
|---|---|---|---|
| | 南京 | 267 | けんぢ |
| | MoAi-NET美唄 | 67 | けんぢ |
| | Network-SCoT | 20 | けんぢ |
| | 梁山泊 | | Kendi. |

lhvは、LZH形式の圧縮ファイルを自動展開するビューワとしてSEEと双璧をなしています。ファイル選択は、通常のファイル名によるものですが、ビューワ機能は検索やエスケープシーケンスへの対応がなされており、スクロールやカーソル移動などのキー操作を自由にカスタマイズすることもできるようになっています。

また、選択したファイルをすべて読み込んでから表示するのではなく、ファイルの最初のほうを読み込んだら、即ビューワモードで表示されるので、大きなファイルを選択したときでも、まったく待たされません。ビューワでファイルの最初のほうを見ている間にも裏で読み込みを続けるという方式です。SEEでも、LZH形式の展開についてはビューワの裏で実行されるようですが、ファイルの読み込みは最初に全部行うようで、lhvと比べると若干待たされます。

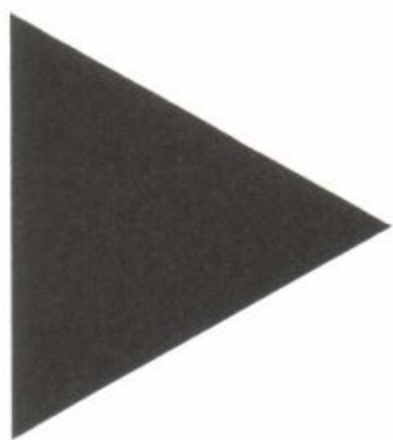
もう１つ、lhvの便利な機能に、LZH形式の圧縮ファイルの中にあるファイルを直接指定する“-f”というオプションスイッチがあります。lhvのドキュメントからの抜粋ですが、コマンドのマニュアル等を１つのLZH形式の圧縮ファイルにまとめておき、以下のようにlhvで引くことができます。

```
lhv  a:/usr/man/man.lzh -f make
```

(これは、man.lzhの中のmakeというファイルを直接指定しています)

以下のようにエイリアスを指定しておけば、"man make"でlhvのビューワがファイルmakeを表示して起動します。

```
alias  man  lhv a:/usr/man/man.lzh -f
```

残念ながら、SEEはLZH形式の圧縮ファイルの中のファイルを直接指定してビューワを起動することはできませんので、ファイル選択画面で起動することになります。ファイル名をソートするようにしておけば、目的のファイルを探すこと自体は、そう難しくはありませんが、一発でマニュアルビューワが起動するlhvのほうがスマートといえるでしょう。

# DC.R

## 超高速、連続複写が可能なディスクコピーツール

**【概要】**……………………

DC.Rは、基本的には通常のDISKCOPY.Xと同じディスクコピーツールですが、その高速性（作者によると、公称理論最速値を達成したということですが）には目を見張るものがあります（筆者が計ってみたところ、ベリファイなしで約2倍の速さでした)。DC.Rは、一度メモリに読み込んでから複写するため（つまり、元のフロッピーディスクを再度読み込むことなく連続複写が可能)、複数のフロッピーディスクに複写することが容易ですし、大量の複写を行う場合も手間がかからず便利です。

また、速度に関しては、FAT（File Allocation Table）を調べ、複写しなくてもよい箇所は読み飛ばして複写するので、その分、複写時間が短縮されます。さらに、画面に処理途中の状況が逐一表示される(現在、どのトラックを読み書きしているかがカラフルに表示される）ので、寂しくなりがちなコマンドラインが華やかになるでしょう。

**【作者】**…………………

| Pop.(伊野敏之) | NIFTY-Serve | GCB02316 |
|---|---|---|
| | サンデーネット | sun2172 |
| | PEKIN | POP. |
| | 梁山泊 | POP |
| | Cecile BBS | CEL0312 |

**【作者からの言葉】**……

これは、ネット上で数多く発表されているディスクコピープログラムでは、わがままな私にとって「まったく不満のないもの」が見つからず、仕方がないので自分で作ってしまったという開発経緯があります。また、私がX68000で作った最初のプログラムでもあります。当時、アセンブラの本を読んで3日目に作ったものがその原型で、それからは多くの類似プログラムの気に入ったところをどんどん取り入れて発展させてきました。手前味噌ですが、自分でもこれはけっこう使えると思っています（他の自作のプログラムに比べてですけれど)。

ネット上で公開したところ、なかなか評判がよく、こんな機能もつけて欲しいといった要望やバグ情報などをたくさん寄せていただきました。おかげで完成度も上がり、機能的にもずいぶんと向上した気がします。そのうえ自分の勉強にもなりましたし（笑)。

その後、コピー以外にフォーマットなどの機能もつけてはどうかという意見をたくさんいただいたのでDC IIの製作に入りました。まだ完成はしていませんけれど……。

DC IIの開発コンセプト(？)は、大げさですが、「高機能と手軽さの追求」です。とにか

く何も考えずに誰でも使えて、必要に応じて高度な利用法もできるようにしたつもりです。まだまだこれからも暇をみて改良していこうと思っていますので、この本を読んでいるあなたも不満点があったら、ぜひパソコン通信を始めて連絡してください。(^_^;)

ちなみに、隠し機能／コマンドがいくつか遊びで入れてあります。気づいたら、どうぞ。最後に、DCを製作する際、資料を提供してくださったMissy.M氏、ならびに多数のデバッグを手伝っていただいたKAPPY.氏に深く感謝します。

**【推薦します】**…………　ディスクコピーはディスクユーザにとってけっこう煩わしいものです。システム付属のものもそんなに遅くはありませんが、数が増えると大変です。しかし、DCならばGRAMを利用しての一気読み、一気書きですので非常に高速です。外付けのドライブにも対応していて、動作も安定しています。レイアウトもよいので、お気に入りです。

MAX BBS　WILLIAMS

**【書式】**……………………DC［-オプション…］［-オプション…］……

**【主な使用法】**…………　通常の起動方法はいたって簡単です。コマンドラインから、

DC[CR]

と入力するのみです。

起動したとき、すでにX68000本体にフロッピーディスクが挿入されている場合は、挿入されているフロッピーディスクをイジェクトします。画面には上下にトラック数が表示され、「ソースディスクを入れて……」という表示が出ます。ここで、読み出しドライブ（デフォルトでは0ドライブ）に複写元のディスクを挿入してください。[CR]キーを押すと、読み出しを開始します。非常に高速に読み出していきます。読み出しを終了すると、複写元のディスクはイジェクトされます。

次に、複写先のディスク（ブランクディスク）を書き込みドライブ（デフォルトでは1ドライブ）に挿入し、[CR]キーを押します。ここでも高速に書き込んでいる様子がわかります。書き込みが終了すると、複写先のディスクもイジェクトされます。イジェクトされると作業は終了です。[ESC]キーを押せば、終了します。ここまでが通常のディスクコピー機能としての使い方です。

最後の[ESC]キーのかわりに[CR]キーを押すと、次のディスクの読み出しに入ります。また、[R]キーを押すと、新たな読み出しは行わずに書き込みだけが行われます。これは、一度読み込んだファイルを複数のディスクに複写する場合に便利です。

**【オプションスイッチ】** ここで指定するもの（起動時点でのオプション）は、DCを起動したあとでも、後述するメニュー画面等で再設定することが可能です。

| オプションスイッチ | 内　　容 | デフォルト |
|---|---|---|
| -C | FATチェックを行わない注1 | FATチェックあり |
| -Dn、n | データを読み出すドライブと書き込むドライブの指定。内蔵ドライブは、0と1注2 | -D0、1 |
| -E | 外部増設ドライブをシャープ純正と同じ取り扱いにする注3 | 違うものとする |
| -F | つねにフォーマットを行う注4 | フォーマットの有無による |
| -G | つねにGRAMを作業領域とする注5 | GRAMの使用状況による |
| -GN | つねにGRAMを作業領域としない | |
| -Hn | ヘッドロード時間注6<br>nには、1から7を指定可能（1=4ms、2=6ms、3=8ms、4=10ms、5=12ms、6=14ms、7=16ms） | -H2 |
| -I | ディスク挿入のチェックなし注7 | チェックあり |
| -K | キーバッファが溜まるようにする | クリアする |
| -L | 緑のアクセスランプの点滅を行わない注8 | 点滅させる |
| -N | ノーマルスピードにする注9 | 高速 |
| -S | シフトフォーマットを行う（コラム参照） | ノーマル |
| -Tn、n | スタートエンドトラック指定注10 | -T0、153 |
| -V | ベリファイチェックを行う注11 | 行わない |
| -? | オプション一覧の表示 | |

注：

1）FATチェックとは、データが書き込まれているかどうかを判断して、書き込まれていない場合、その箇所をスキップして読み出すようにすること。

2）データを読み出すドライブと書き込むドライブを同じドライブにすると、1つのドライブでディスクコピーが可能。

3）純正として取り扱う場合は、ディスクイジェクトさせたり、LEDを点滅させたりする。

4）通常は複写先のディスクがフォーマットされているかどうかを判断し、フォーマットされていない場合は自動的にフォーマットを行う。しかし、"-F"オプションをつけると、それを無視してすべての場合（つまり、すでにフォーマットされている場合も）、フォーマットを行うようになる。また、よく行われるシフトフォーマット（コラム参照）については、フォーマットしていないものと判断するので注意が必要。

5）通常、GRAMは他の目的（RAMディスク等）に使用されていない場合にのみ作業領域として使用する。このオプションをつけると、GRAMをRAMディスクとして使っている場合はRAMディスクの内容が破壊される可能性があるので注意が必要。通常は、このオプションをつける必要はない。

6）ヘッドロード時間はディスク読み出しの待機時間なので、短くすれば、それだけ速く複写できる。しかし、速くすればするほど、リードエラーを起こしやすくなるので、あまりエラーが多発するようなら、遅めに設定すること。

7）通常は、ディスクの挿入状態を判断して読み出し動作を行うようになっている。ただし、キー入力は必要。また、複数のディスクを連続して読み出す場合、読み出しおよび書き込みの動作の直前にキー入力を求めるようになっている。このオプションを指定することにより、挿入状態を判断せずに、すぐに読み出し動作を行うようになる。また、連続して読み出す場合、読み出しおよび書き込み動作の直前のキー入力を求めないようになる。MENUを終了した場合も、キー入力を待たない。ただし、最初の読み出しに関しては、キー入力が必要。

8）通常はディスクの挿入を催促するために、ドライブランプを点滅させる。
9）絶対にエラーを起こしてはいけない場合等に指定する。
10）DCでは、コピートラックを指定することが可能。指定可能なトラックは0から165まで。ただし、通常は指定する必要はない。
11）書き込みの確認（ベリファイ）を行う。確実に複写されたかどうか確認する場合は、これを指定する。ただし、この指定をすると、コピースピードは2倍近く遅くなる。

COLUMN

**シフトフォーマット**

通常、フォーマットする場合は、磁気ディスク上のセクタの始まりを一直線に揃えてフォーマットします。外部記憶装置（磁気ディスク、ハードディスク、フロッピーディスクドライブ、MO等）のヘッドの移動速度が高速な場合にはこれで問題は起こらないのですが、磁気ディスクヘッドの移動が遅いフロッピーディスクドライブなどの場合には、セクタの始まりを逃してしまうことがあります。そのような事態にならないように、セクタの始まりを少しずらして並べると、セクタの始まりを逃すことが少なくなり、外部記憶装置へのアクセスも速くなります。このようなフォーマットを、「シフト（正確にはセクタずらし）フォーマット」といいます。

**【環境設定】**

DC用の環境変数としては、dc2_optがあります。この環境変数には、通常使用するオプションを設定しておくことができます。

例）

```
set dc2_opt=-C -F -V
```

**【操作方法・キー一覧】**

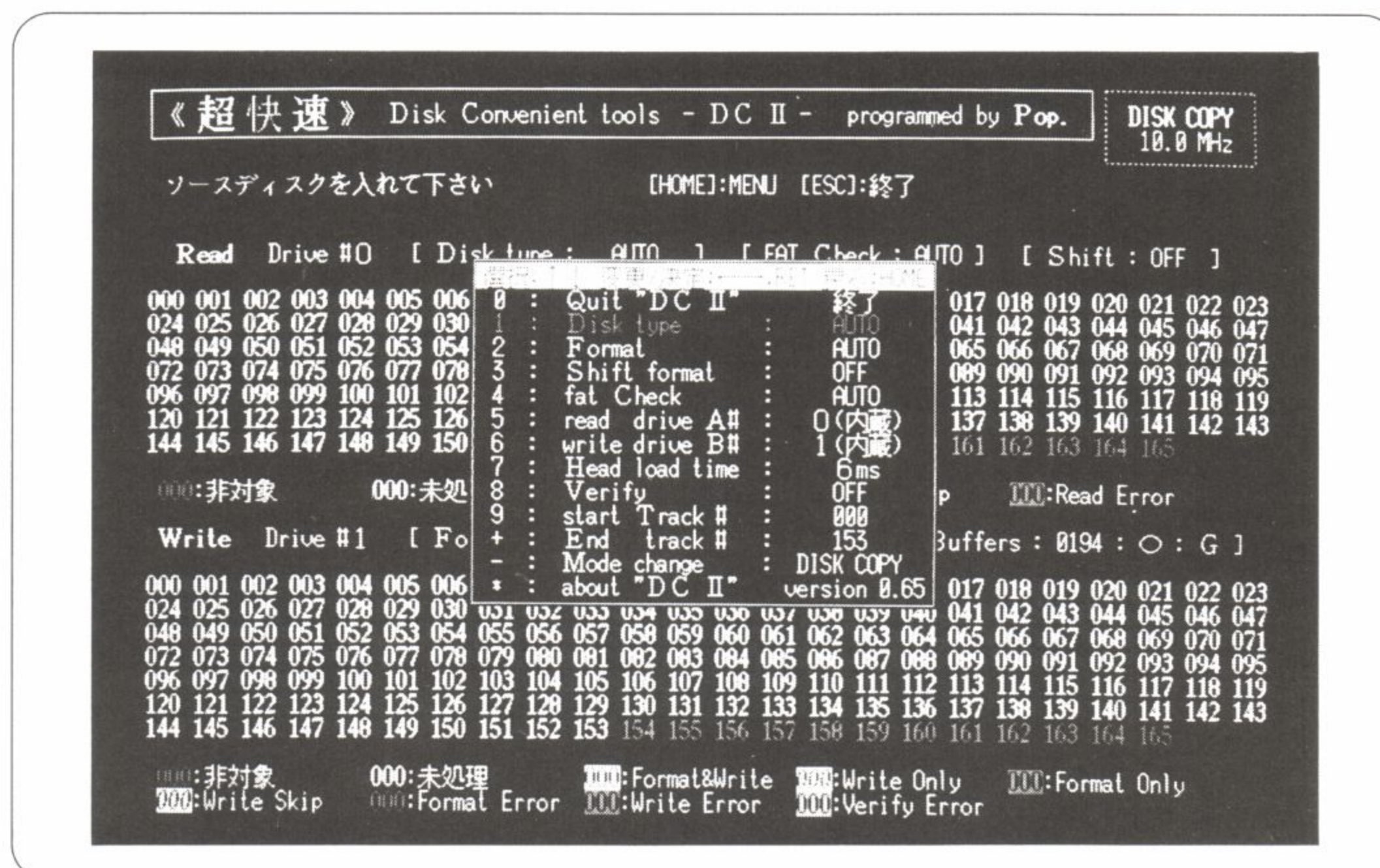
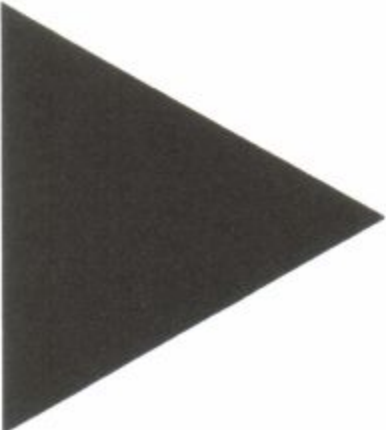
## 1. 通常画面における操作

DCの動作している最中に次のキーを押すことにより、動作を随時変更することが可能です。

| 操作キー | 動　　作 |
|---|---|
| [ESC] | DC の終了 |
| [HOME]、[HELP]、[CLR] | 環境設定メニューの起動、およびメニューの終了 |
| [CR] | 作業の継続 |
| [0] | 読み出し（複写元）ドライブのディスクイジェクト |
| [1] | 書き込み（複写先）ドライブのディスクイジェクト |
| [F] | 物理フォーマットの有無 |
| [S] | シフトフォーマットの有無 |
| [C] | FATチェックの有無 |
| [A] | 読み出しドライブの設定 |
| [B] | 書き込みドライブの設定 |
| [H] | ヘッドロード時間の設定 |
| [V] | ベリファイチェックの有無 |
| [R] | 別のディスクに書き込む |
| [T] | スタートトラック指定 |
| [E] | エンドトラック指定 |

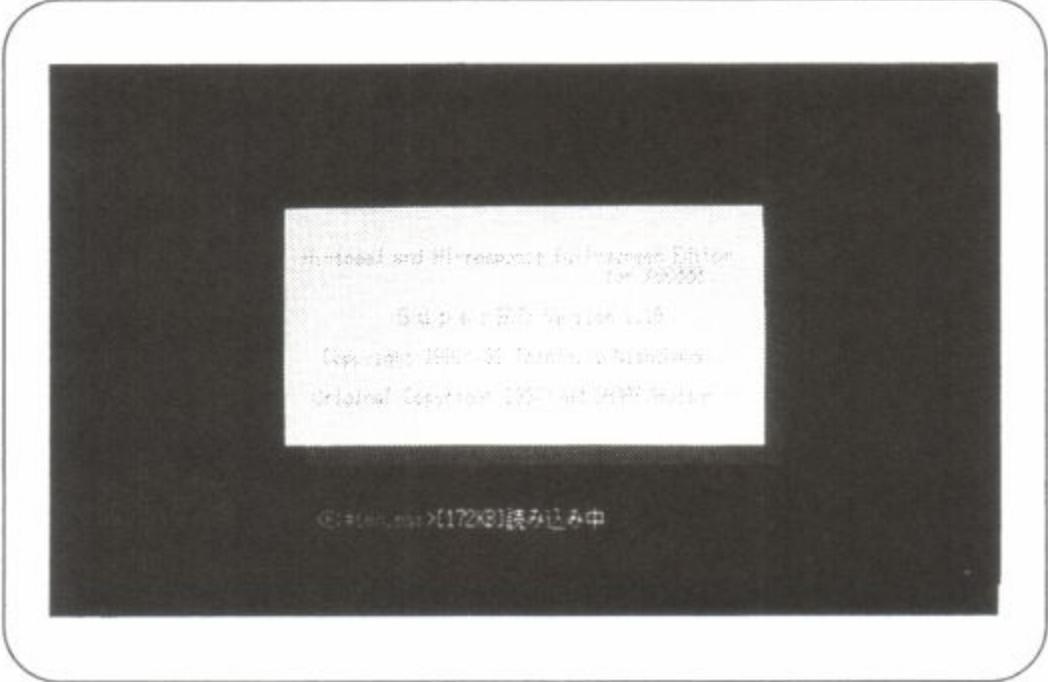
## 2. 環境設定メニュー画面における操作

[HOME]、[HELP]、[CLR]など、いずれかのキー操作により、環境設定メニュー画面になります(Pht.1参照)。

Pht.1 DCのメニュー画面

[↑][↓]キーにより項目を選択して、[←][→]キーにより変更を行い、[CR]キーで決定します。これを繰り返して設定が終了したら、[HOME]、[HELP]、[CLR] キーを押せばメニューを終了します。

# SUPERED.X

## 超高速、高機能なED.Xコンパチブルエディタ

【概要】……………………　X68000に付属しているED.Xを、大幅に高速化、機能強化したフルスクリーンエディタです。起動、スクロール、置換、検索、……すべてがED.Xと比べて高速化されており、機能面では文字単位のカット&ペーストが可能です。また、編集ファイルを指定しないで起動した場合、ファイル選択が可能なファイラ機能、スムーズスクロール（[OPT.1]+[ROLL UP]、[ROLL DOWN]）など、豊富な機能が盛り込まれています。また、PC-9801で使われている、俗に言われる「98文字」を表示することができます。

Pht.1　ED.Xの編集画面(左)とSUPERED.Xの編集画面(右)

注：SUPERED.XがED.Xに比べてどのくらい高速かを、200個の単語の置換速度と、[ROLL UP]キーでの1000行のスクロール速度とで比較してみたところ、以下のようになりました。

計測機種：X68000 PRO（10MHz）にて計測

| | 単語200個の置換 | [ROLL UP]キーで1000行のスクロール |
|---|---|---|
| ED.X | 12秒 | 36秒 |
| SUPERED.X | 5秒 | 27秒 |

（手元の時計で計測）

以上のように、少なくとも約2倍の高速化がなされているようです。

【作者】……………………サンデーネット　SUN0064　devil
MAX BBS　MAX0064　OH!

【作者からの言葉】……　数々のバージョンアップを経てSuperEDは現在の形となりましたが、当初から「あまり趣味に走らない」ということを念頭に、やってきました。また、装飾機能よりも基本性能を重視し、使っていてイライラしない、気持ちよいレスポンスを大切にしてきたつもりです。したがって、見かけは地味ですが、SuperEDを使っていただければ、その気持ちのよ

さを肌で感じとっていただけると思います。このエディタを何に使うかは、みなさんの自由ですが、このエディタを使って、たくさんの優秀なソフトウェアが生み出されればうれしい限りです。

**【推薦します】**

SuperEDの滑らかスクロールは重宝します。使い出すともう滑らかじゃないと体が受け付けません。あと、この行の、この一部分をコピーしたいなーとかいうときも便利です。基本的に標準添付のEDの機能は持っていますので、EDからの移行も楽だと思います。と、くれば、もうこれは使うしかないでしょう！

MAX BBS　てぃん

X68000にあるエディタの中で比較的使いやすく、普通に使うぶんには何も不満は感じない。文章やアセンブラのソースなど、すべてこのエディタだけで足りてしまう。tfやFuなどと組み合わせて使うと、使いやすさはさらに向上するだろう。

MAX BBS　でぃがん

**【使用法】**

SUPERED［オプションスイッチ］［ファイル名］

例）

```
SUPERED /P0,10,0,0,0 /N1 /X1 SAMPLE.DOC *.TXT
```

上記の例の場合だと、オプションスイッチは、テキスト画面の黒の部分を緑に変更して(/P0,10,0,0,0)、バックアップファイルを作らず(/N1)、カーソルが桁位置を記憶している(/X1)というモードで、編集ファイル名は、SAMPLE.DOCと＊.TXTという複数のファイルを呼び出して起動させています。編集ファイル名指定の最大数は10個です。

編集ファイル名を省略してSUPERED.Xを起動させると、ファイラが起動します。

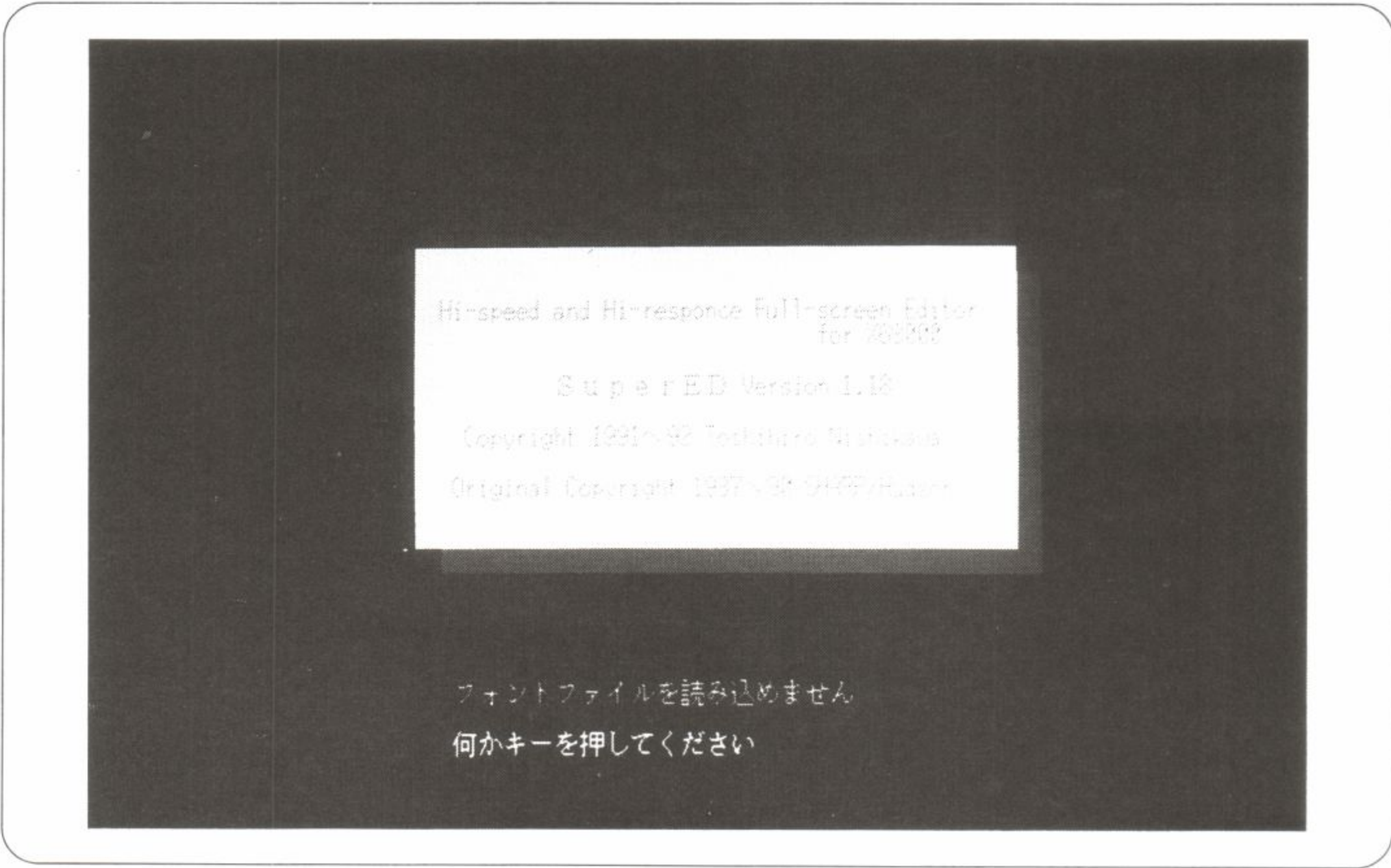
**【起動時の注意】**

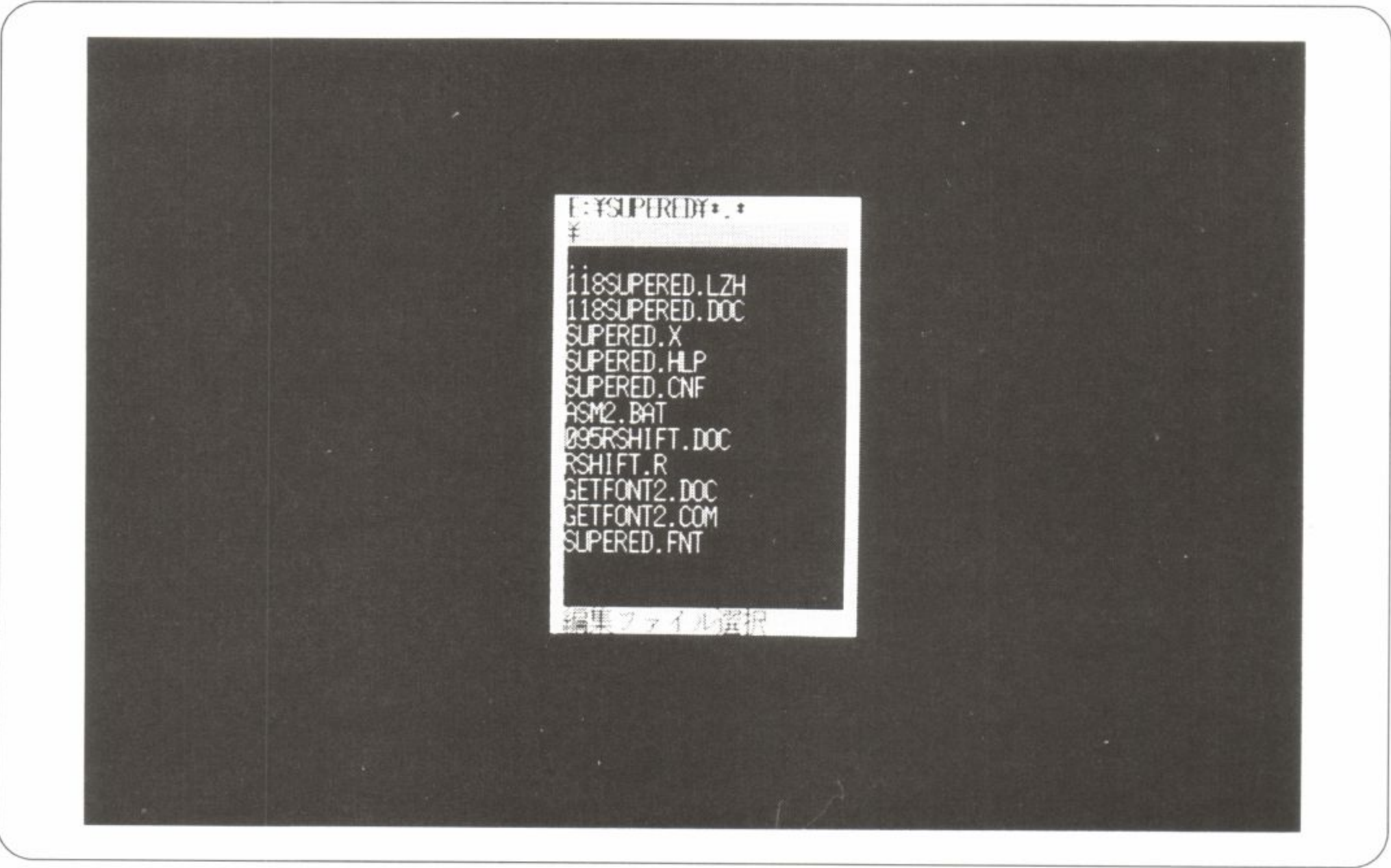
SUPERED.Xは、118SUPERED.LZHを解凍して、そのままSUPERED[CR]として起動させても、216ページのPht.2のようになるだけで、起動しません。この場合は、[OPT.1]キーを押しながら起動するか、118SUPERED.LZH内にあるSUPERED.CNFファイルを編集するか、GETFONT2.COMを使用してSUPERED.FNTを作成してください。

ここでは、SUPERED.CNFを書き換えてSUPERED.Xを使用する方法を説明します。SUPERED.CNFの中の、

```
%FNT_FIL ="B:¥FON¥SUPERED.FNT"        * 拡張フォントファイル名
```

という行を削除するか、行頭に＊をつけてコメントアウトしてください。そのあと、SUPERED[CR]と起動させると、起動画面が現れ、ファイラが表示されます。

Pht.2　「フォントファイルを読み込めません」というメッセージ



Pht.3　ファイラ画面

なお、SUPERED.FNTを作る方法についてはドキュメントを参照してください。

SUPERED.FNTができあがったら、SUPERED.CNF内の、

%FNT_FIL ="B:¥FON¥SUPERED.FNT"　　　＊拡張フォントファイル名

という行を、SUPERED.FNTが実際に置いてあるディレクトリに変更します。

この2つのうち、どちらかの方法を実行すると、SUPERED.Xは起動します。

【主な機能】

SuperEDのキー操作は、システム付属のED.Xと基本的に互換性があります。しかし、SUPERED.Xは機能が拡張されているため、いくつかのキー操作が追加されています。

以下、SuperEDの主な機能を紹介します。その他のキー操作についてはSUPERED.HLPを参照してください。追加されたキー操作は、[OPT.1]+[]、[SHIFT]+[]です。

### 1. コピーバッファ機能

コピーバッファ機能とは、ファイル上の任意の文字または行をコピーバッファへ取り込み、あとからその内容をファイルの任意の場所へコピーできるという機能です。

コピーバッファへの取り込みは、[OPT.1]+[UNDO]ででき（1文字単位でコピーできます)、コピー（貼り込み）は[OPT.1]+[XF1]でできます。

コピーバッファの消去は、[OPT.1]+[XF2]です。

### 2. 行番号メモリバンク機能

行番号メモリバンク機能とは、任意の行を登録し、その登録した行へジャンプできる機能です。メモリバンクは5つあり、それぞれの登録方法は下のようになっています。

Tbl.1 メモリバンクキー割り当て表

| 種類 | 登録 | 消去 | 移動 |
|---|---|---|---|
| メモリバンク1 | [OPT.1]+[F1] | [OPT.1]+[F6] | [OPT.1]+[1] |
| メモリバンク2 | [OPT.1]+[F2] | [OPT.1]+[F7] | [OPT.1]+[2] |
| メモリバンク3 | [OPT.1]+[F3] | [OPT.1]+[F8] | [OPT.1]+[3] |
| メモリバンク4 | [OPT.1]+[F4] | [OPT.1]+[F9] | [OPT.1]+[4] |
| メモリバンク5 | [OPT.1]+[F5] | [OPT.1]+[F10] | [OPT.1]+[5] |

注：[1]〜[5]はテンキーではなく、[1]〜[5]のフルキーを指す。

また、メモリバンクへの登録状況は画面右下の「空」という文字の右側、横5本線のマーカーによって確認できます。マーカーは上のラインから順にメモリバンク1〜5となっており、何も登録されていない状態では水色、登録されている状態では黄色となっています。

**注意**

1）すでに登録されているメモリバンクは、誤って上書きしてしまわないように、いったん消去しないと再登録できないようになっている。登録を消去せずに新たに登録しようとすると、BEEP音を鳴らして警告してくる。

2）メモリバンクへ登録されるのは、その行の内容などではなく、行番号である。編集中のファイルへ挿入や削除を行うと、必ずしも以前登録した箇所にジャンプするわけではない。

## 3．オートアセンブル機能

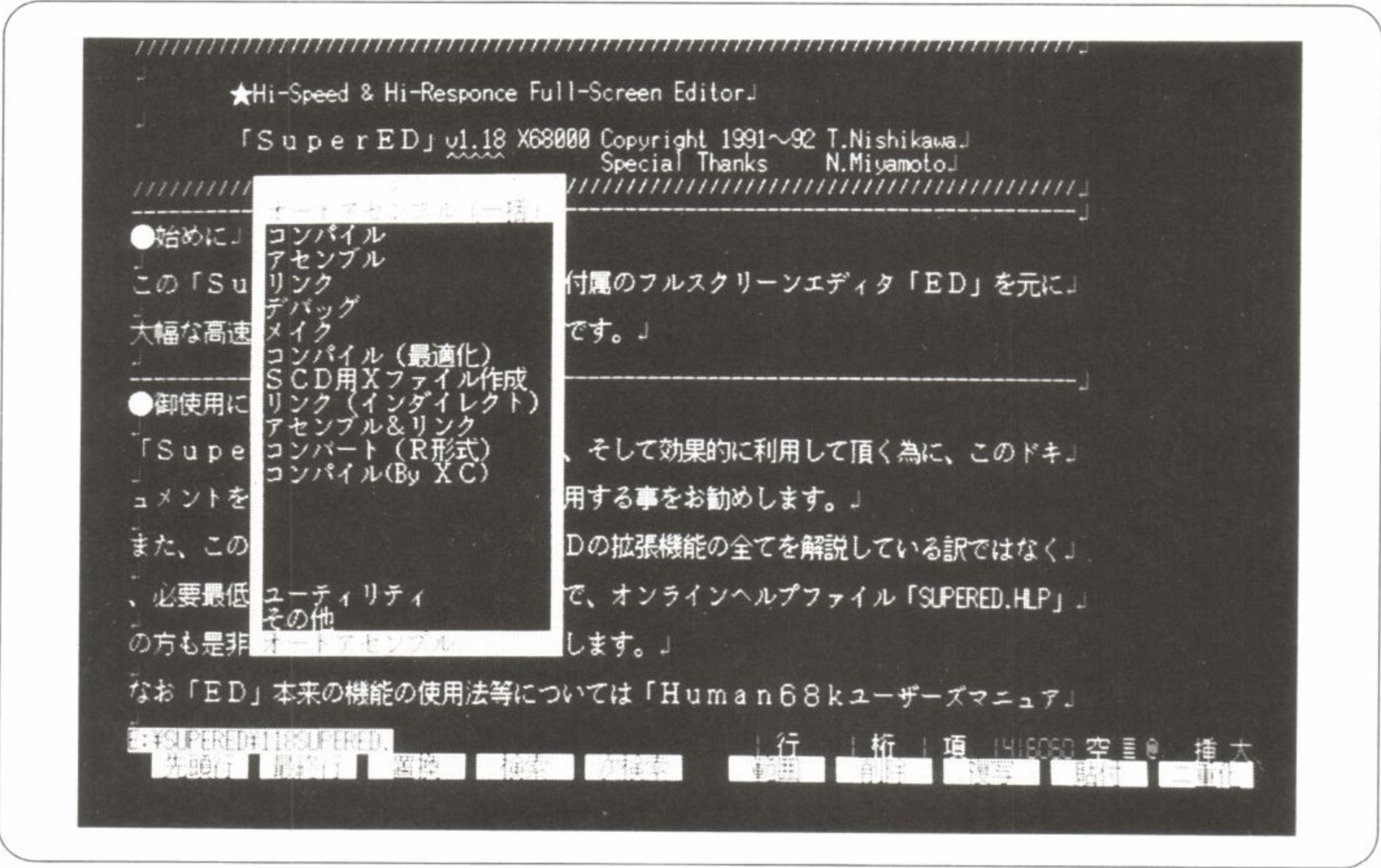
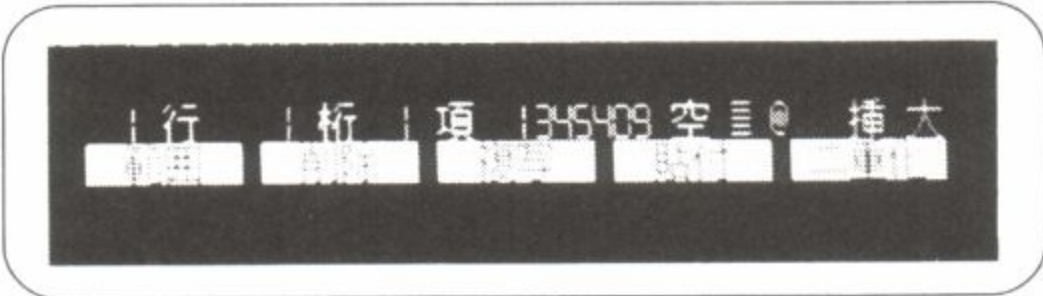
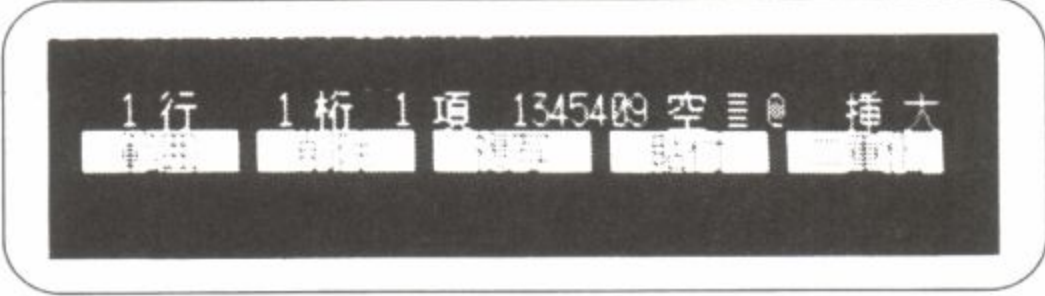
オートアセンブル機能とは、SUPERED.Xを終了させずに現在編集中のソースファイルのコンパイル、アセンブル、リンクなどが実行できる機能です。同梱のASM2.BATを編集することにより実行可能となります。この機能を使う場合は、[OPT.1]＋[XF4]を押すとメニューが出てきますので（ASM2.BATの場合）、その画面から実行してください。

使用する際は“SUPERED.CNF”を編集します。

まず、SUPERED.CNF(223ページFig.1)の、

```
%ASM_CMD = "A:¥BIN¥ASM2.BAT"        ＊アセンブルコマンドファイル名
```

という行を、実際に“ASM2.BAT”があるディレクトリに変更します。そのうえで[OPT.1]＋[XF4]キーを押すと、起動します(Pht.4)。

**Pht.4**　オートアセンブルのメニュー画面

## 4．ファイラ

ファイラとは、起動時にファイル名を入力せずに、ファイル一覧の中からファイルを選

択できる機能をいいます。カーソルキーの[↑][↓]キーでファイルを選択し、[CR]キーで決定。[→][←]キーでドライブを変更することができます。

**【オプションスイッチ】** オプションスイッチを指定しない（デフォルトの）場合、以下の[　]（ブラケット）でくくられたものが指定されます。

| オプションスイッチ | 内　容 |
|---|---|
| -? | コマンドラインヘルプ<br>コマンドラインヘルプを表示する |
| /Aa,b,c | スクロールモード関連スイッチ<br>a　[OPT.1]＋[ROLL UP]または[ROLL DOWN]の操作時のスクロールモードを指定する<br>[0]：スムーズスクロール<br>1：任意行スクロール<br>b　任意行スクロールを指定したときのスクロールする行数を指定する<br>1〜[15]〜30：スクロール行数指定<br>c　スムーズスクロールを指定したときのスクロールスピードを指定する<br>[0]：低速<br>1：高速 |
| /Ba | ファイル読み込みモード関連スイッチ<br>a ファイル終端判別モードを指定する<br>[0]：エンドコード($1A)を終端と認識<br>1：物理的なファイルの終わりを終端と認識<br>($1Aを通常のコードとする) |
| /Ca,b,c | 範囲指定表示関連スイッチ<br>a　[F6]キーを押したときの範囲指定表示を指定する<br>0：指定中の範囲を表示しない<br>[1]：指定中の範囲を表示する<br>b　表示タイプを指定する<br>[0]：反転<br>1：網かけ<br>c　範囲指定中にパレット3（白）の明るさを落とす度合いを指定する<br>[0]〜31：パレットを落とす度合い(31が一番暗くなる。一般的には8が適当) |
| /Da,b | パラメータ数字関連スイッチ<br>a　パラメータ表示（日本語変換行の上の行）用の数字フォントを指定する<br>[0]：デジタルフォント（Pht.5）<br>1：IOCSフォント（Pht.6）<br>b　パラメータ表示用の数字フォントの色を指定する<br>1、[2]、3：表示色（1＝水色、2＝黄色、3＝白） |

**Pht.5** デジタルフォント

**Pht.6** IOCSフォント

| | |
|---|---|
| /Ea,b,c | ファイル終端記号関連スイッチ<br>a　ファイル終端記号（[EOF]）の表示の有無を指定する<br>　0 ：表示しない<br>　[1]：表示する<br>b　ファイル終端記号の表示色を指定する<br>　[1]～3 ：表示色（1＝水色、2＝黄色、3＝白）<br>c　ファイル終端記号のフォントを指定する<br>　0～[1]～4 ：フォント番号 |
| /Fa,b,c | アクセサリ関連スイッチ<br>a　時計の表示を指定する（[OPT.1]＋[CLR]）<br>　[0]：表示しない<br>　1 ：表示する<br>b　スケールメータの表示を指定する（[OPT.1]＋[4](テンキー)）（Pht.7）<br>　[0]：表示しない<br>　1 ：表示する<br>c　オートアセンブル終了ブザーの使用を指定する<br>　[0]：鳴らさない<br>　1 ：鳴らす |

**Pht.7**　スケールメータ

| | |
|---|---|
| /Ga,b | フォント関連スイッチ<br>a　全角スペースを実体化した際のフォントを指定する<br>　（実体化：[ESC]・[I]）<br>　[0]～5 ：フォント番号<br>b　シフトJISコード$ED40～$EEFCの領域に表示するフォントを指定する<br>　[0]：98拡張漢字（IBMフォント）<br>　1 ：68拡張外字 |
| /Ha,b,c,d | ファイラ関連スイッチ<br>a　起動時にファイラでのファイル選択にするか、手動入力にするかを指定する<br>　0～[6]～7 ：起動優先順位注<br>b　SuperED起動時のファイル選択モードを指定する<br>　[0]：シングル選択モード<br>　1 ：マルチ選択モード（最大10ファイル）<br>c　選択確定時にウィンドウカーソルを点滅させる回数を指定する<br>　0～[3]～10 ：点滅回数（0 にすると点滅なし）<br>d　選択確定時のウィンドウカーソルの点滅周期を指定する<br>　0～[5]～20 ：点滅周期（数字が小さくなるほど点滅周期が短くなる） |

| | |
|---|---|
| /Ia,b,c | カーソルリピート関連スイッチ<br>a　[OPT.1]+[←][→]のリピートディレイ（キーを押し続けたときの処理）を指定する<br>0〜[1]〜10　：ディレイ(0を指定すると速くなり、10だと遅くなる)<br>b　[←][→][↑][↓]の1回目のリピートディレイを指定する<br>0〜[25]〜100　：ディレイ<br>c　[←][→][↑][↓]の2回目以降のリピートディレイを指定する<br>0〜[2]〜10　：ディレイ |
| /Ja,b,c | 桁位置指示ライン関連スイッチ<br>a　桁位置指示ラインの表示を指定する（[OPT.1]+[7](テンキー)）<br>[0]：表示しない<br>1　：表示する<br>b　桁位置指示ラインの表示桁位置を指定する<br>0〜[80]〜94：桁位置<br>c　カラムゲージの表示を指定する([OPT.1]+[G])<br>[0]：表示しない<br>1　：表示する |
| /La,b,c | 改行記号関連スイッチ<br>a　改行記号の表示の有無を指定する<br>0　：表示しない<br>[1]：表示する<br>b　改行記号の表示色を指定する<br>1〜[2]〜3　：表示色（1=水色、2=黄色、3=白）<br>c　改行記号のフォントを指定する<br>[0]〜13　：フォント番号 |
| /Ma,b | 行バッファ関連スイッチ<br>a　1行のバッファサイズ(単位はバイト)を指定する<br>128、256、512、[1024]：バッファサイズ<br>b　テキストエリアへの挿入、削除に伴うデータ転送の方式を指定する<br>[0]：MPU転送<br>1　：DMA転送<br>（最大速度。転送中バスを独占する）<br><br>補足：所要時間10（MPU）対7（DMA）ではDMA転送のほうが速い（10MHz時）。ただし、DMAは最大速度で動かしているため、DMA転送中にMPUによるメモリアクセスはできなくなる |
| /Na | バックアップモード関連スイッチ<br>a　バックアップファイルの作成を指定する<br>[0]：作成する<br>1　：作成しない |
| /Pa,b,c,d,e | テキストパレット関連スイッチ<br>a　変更するパレット番号を指定する<br>0〜3　：テキスト画面（0=黒、1=水色、2=黄色、3=白）<br>4〜5　：マウス、キーボード、電卓画面（4=茶灰 5=紺）<br>b　G(緑)の値を指定する<br>0〜31：緑の度合い<br>c　R(赤)の値を指定する<br>0〜31：赤の度合い<br>d　B(青)の値を指定する<br>0〜31：青の度合い<br>e　I(輝度)の値を指定する<br>0〜1：輝度の度合い |

| | |
|---|---|
| /Ra | キー割り込み関連スイッチ<br>a　キー割り込み独占の有無を指定する<br>[0]：独占しない<br>1　：独占する（IOCSレベルに限る。ROMアドレスに依存）<br><br>補足：<br>0：外部のキー割り込みを変更して常駐するプログラムを利用したい場合に指定する<br>1：外部のキー割り込みを変更して常駐するプログラムを利用できないようにしたい場合に指定する |
| /Sa | 画面モード関連スイッチ<br>a　画面モードを指定する<br>64　：64桁モード<br>[96]：96桁モード |
| /Ta.b,c | タブ関連スイッチ<br>a　タブ記号および全角スペース記号表示の有無を指定する<br>[0]：表示しない<br>1　：表示する<br>b　タブ記号の表示色を指定する<br>[1]～3　：表示色（1＝水色、2＝黄色、3＝白）<br>c　タブサイズを指定する<br>2、4、6、[8]：タブサイズ |
| /Xa,b | カーソル移動モード関連スイッチ<br>a　カーソル桁位置記憶モードを指定する<br>[0]：非記憶モード<br>（カーソル桁位置は行の長さに左右される）<br>1　：記憶モード<br>（カーソル桁位置は一時的にしか行の長さに左右されない）<br>b　オートインデントモードを指定する<br>[0]：通常モード（通常の改行が行われる）<br>1　：オートインデントモード(改行後に自動的にインデントが行われる) |

注：スイッチで指定する0～7とは、bit0、1、2からなる2進数を10進数変換したものを設定する。
例を挙げて説明すると、

例）
bit2＝on(1)　bit1＝on(1)　bit0＝off(0)　→　110(2進)→　6(10進)

各bitは、on(1)のときはファイラ選択モードになり、off(0)のときは手動選択モード(ED.Xのときと同じように編集するファイルネームを打ち込む必要がある）という意味になる。以下、各bitの説明をしておく。

bit2は、新規ファイルを編集するときにどうするのかを聞いてくる。on(1)のときは新規ファイルの編集時にファイラが起動し、off(0)のときは手動入力モードになる。

bit1は、ファイル読み込み([ESC]＋[F]、[SHIFT]＋[F1])のときにどうするのかを聞いてくる。on(1)のときはファイル読み込み時にファイラが起動し、off(0)のときは手動入力モードになる。

bit0は、ファイル書き出し([ESC]＋[W]、[SHIFT]＋[F9])のときにどうするのかを聞いてくる。on(1)のときはファイル書き出し時にファイラが起動し、off(0)のときは手動入力モードになる。

例のように"1,1,0"と入れたときは、「新規ファイル、ファイル読み込み時はファイラ起動、

ファイル書き出し時は手動入力」という意味になる。

そして、110という2進数を10進数に変換すると「6」になるので、/Haの“a”に「6」を入れればよいことになる。

**注意**

SUPEREDには上記のオプションスイッチや、フォントや読み込みファイル等の各種のファイルを登録するためのコンフィギュレーションファイル“SUPERED.CNF”があり、このファイルにあらかじめ設定しておくことができます。

**Fig.1** コンフィギュレーションファイル"SUPERED.CNF"

```
*《 コンフィギュレーションファイル for ＳｕｐｅｒＥＤ Version 1.18 》

*このファイルは「ＳｕｐｅｒＥＤ」の環境を設定する為のものです。
*自分の環境に合わせこのファイルを書き換えてください。
*このファイルの設定に問題があり「ＳｕｐｅｒＥＤ」の起動に失敗した場合は [OPT.1]
キーを押しなが
*らＳｕｐｅｒＥＤを起動しコンフィグファイルを修正してください。（[OPT.1]でCONFIG
がパスされる）
*＜書式など＞
*１．「%」はコマンドヘッダで、その直後の文字をコマンドとして解釈します。
*２．データはコマンドの右側に記述し、必ず”～”のように「”」で囲んでください。
*３．「*」以降のデータは行末まで全て注釈として解釈します。
*４．「%ERR_TAG」はパスを省略すると自動的にテンポラリパス（temp）下となります。
*５．使用しないコマンドは頭に「*」を付けるなどして効かないようにしてください。

%SWITCH  = "/P0,5,2,10,0 /T1,1,8 /H6,0,0,0"     *スイッチ登録
%ASM_CMD = "A:¥BIN¥ASM2.BAT"            *アセンブルコマンドファイル名
%ERR_TAG = "AE.TAG"                     *アセンブルエラータグファイル名
%FNT_FIL = "A:¥BIN¥ED.FNT"              *拡張フォントファイル名
%HLP_FIL = "A:¥BIN¥ED.HLP"              *オンラインヘルプファイル名
%MAC_FIL = "A:¥BIN¥SUPERED.MAC"         *キーボードマクロ定義ファイル名
%INC_PAH = "B:¥INC"                     *インクルードパス名

%ASM_MNU = "0,オートアセンブル（一括）" *#オートアセンブルメニュー用データ
%ASM_MNU = "1,コンパイル"
%ASM_MNU = "2,アセンブル"
%ASM_MNU = "3,リンク"
%ASM_MNU = "4,デバッグ"
%ASM_MNU = "5,メイク"
%ASM_MNU = "6,コンパイル（最適化）"
%ASM_MNU = "7,ＳＣＤ用Ｘファイル作成"
%ASM_MNU = "8,リンク（インダイレクト）"
%ASM_MNU = "9,アセンブル＆リンク"
%ASM_MNU = "10,コンバート（Ｒ形式）"
%ASM_MNU = "11,コンパイル(By ＸＣ)"

%ASM_MNU = "16,ユーティリティ"
%ASM_MNU = "17,その他"
```

このファイルを編集作成しておくと、次回起動時からオプションスイッチを省略することができます。なお、SUPERED.CNFの設定に失敗して起動できないこともあるかと思います。そのような場合は、[OPT.1]キーを押しながらSUPERED.Xを起動してください。SUPERED.CNFで設定されたスイッチを無視してSUPERED.Xが起動します。この機能を利用してSUPERED.CNFを編集してください。また、SUPERED.CNFはSUPERED.Xと同じディレクトリに入れておけば、自動的に読み込まれますが、違うディ

レクトリに入れる場合やSUPERED.XやSUPERED.CNFのファイル名を変更した場合は、環境変数“SED_CNF”に登録してください。

**【その他】**……………… この他にも、ED.Xコンパチブルエディタとしては、ED.R（作者：RANNさん）や、TED.X（同名作品多数）などがあります。

# tsort.r

## 数字を認識できるディレクトリsortコマンド

【概要】…………………… tsort.rは、ディレクトリ上のファイル名の並びをソーティングするプログラムです。また、ファイル名末尾の数字をもきちんと認識してくれます。

通常のソートプログラムでは、

```
cmbx9.tfm
cmbx10.tfm
```

のような桁数のあっていないファイルをソートした場合、

```
cmbx10.tfm
cmbx9.tfm
```

とソートされてしまいます。

しかし、tsort.rでは、桁単位ではなく、数字の部分は数の大小に注目して、

```
cmbx9.tfm
cmbx10.tfm
```

のようにソートしてくれます。

また、オプションスイッチをつけることにより、ファイルサイズやタイムスタンプ、大文字・小文字を区別したソートもできます(もちろん、オプションを複数指定したソートもできます)。

【作者】…………………… SHUNA

| | |
|---|---|
| 梁山泊 | SHUNA |
| Cecile-BBS | CEL0069 |
| KAZUKUN-NET | KAZ0002 |
| なりゆきネット | SHUNA |

【作者からの言葉】…… はじめまして、SHUNA (しゅな) です。最近は仕事も忙しくて、なかなか趣味のプログラムを作れません。そのわりに速いマシンを欲しがっていますが……。このプログラム

の最初のバージョンは、カレントディレクトリを普通にソートするだけで、他の機能は何にもついてませんでした。その後、追加した機能の多くは、使ってくれた人からのレポートに基づいています。フリーソフトを入手したら、希望とか気になったところ、バグと思われるものがあれば、積極的にレスポンスを返しましょう。フリーソフトの作者側としても、こーいうレスポンスはけっこううれしいものです。で、作者が改良を加えて、さらに発展していくというのが、この世界の正しい状態です。

68を買ったからには、みぃんな趣味のプログラマーと化して欲しいですね。68ほどプログラムが素直に組めるマシンは（パソコンレベルでは）他にないし。私なんか、仕事でも68使ってるぐらいです。「家族断絶」とか、「離婚」とか、「そんな息子に育てた覚えはない」とか、「出てって！」とか、「あんたなんか嫌いよ！」とかにならない程度に、みなさん精進してください。

**【推薦します】**…………“DIR”は最も多く使われるコマンドであるといわれますが、その出力は不規則に並んでおり、決して見やすいとはいえません。そこでtsortが必要になります。tsortは、ファイル名末尾の数字をきちんと認識してくれます！　12が2より大きいことがわかってくれるソートプログラムは意外と少ないのです。

MAX BBS　Zeno

**【使用法】**………………tsort -[オプションスイッチ][ドライブ名：ディレクトリ名]

**【オプションスイッチ】**

| オプションスイッチ | 内　　容 | デフォルト |
|---|---|---|
| -s | サブディレクトリがあれば、その内部もソートする | ソートしない |
| -t | タイムスタンプでソートする | (無視) |
| -f | ファイルサイズの小さい順にソートする | (無視) |
| -e | 拡張子優先でソートする | ファイル優先 |
| -n | 数字を認識しない | 認識する |
| -c | 大文字、小文字を識別する（大文字優先） | 識別しない |
| -d | サブディレクトリをファイルの後ろにしてソートする | ファイルが後ろ |
| -r | 逆順でソートする | 普通にソート |
| -z | 環境変数、tsort、tsortext を無視してソートする | 参照する |

**注意**

オプションスイッチは“-”だけが有効です。“/”は、認識されません。また、Human68k Ver1.?? には対応していませんし、TwentyOne.xのマルチピリオドにも対応していません（3文字を超える拡張子を持つファイルは、順番がどうなるかわかりません。ファイルが消えるようなことはありませんが）。

**【主な使用法】**…………　tsortでは、tsortとtsortextの2つの環境変数が使えます。以下、それぞれの環境変数の違いを説明しておきます。

・tsort

オプションスイッチをあらかじめ設定しておけます。設定する場合は、AUTOEXEC.BAT内で、

```
set tsort=scd
```

としておくと、コマンドラインでのオプションスイッチの設定は必要なくなります(上記の設定の場合は、s=サブディレクトリもソートし、c=大文字・小文字の区別をし、d=サブディレクトリをファイルの後ろにソートさせるモードの指定をしています)。

これに対し、コマンドラインでオプションスイッチを指定した場合は、環境変数で設定しておいたスイッチは無視されます。環境変数で指定した設定を一時的に変更したい場合には、コマンドラインから指定してください。

また、環境変数"tsort"に間違ったスイッチを設定してしまった場合、設定を直さない限り動作しません。AUTOEXEC.BATを再編集してください。

例)

```
set tsort=q
```

上記の例ですと、無意味なオプションスイッチの指定ですので、tsortは起動しません。

・tsortext

tsortextを使うと、拡張子が違うファイルをソートしたい場合、あらかじめ自分が並べたいと思っている拡張子の順番を自由に設定できます。つまり、環境変数"tsortext"が設定されている場合は環境変数を無視するオプション"-z"が指定されていない限り、必ず"tsortext"は参照され、ソート中の画面に"tsortext 参照"と表示されます。

設定する場合は、たとえば、autoexec.bat内で、

```
set tsortext=c;s;doc;tex;r;x;z
```

と設定すると、ソース (c、s)、ドキュメント (doc、tex)、実行プログラム (r、x、z) の順番に並べてくれます。また、設定されていない拡張子を持つファイルは、その後ろになります。

例)tsort255.lzh内のファイル＋αを上記の設定で並べ替える場合

| | | | | |
|---|---|---|---|---|
| tsort | doc | 8670 | 92-02-14 | 3：46：44 |
| tsort | r | 4516 | 92-02-14 | 3：04：04 |
| tsort | s | 29257 | 92-02-14 | 3：03：46 |
| tsort | tex | 10031 | 92-02-14 | 4：43：38 |
| test | doc | 6802 | 93-01-19 | 1：44：02 |
| test | r | 6802 | 93-01-19 | 1：44：02 |
| test | tex | 6802 | 93-01-19 | 1：44：02 |

とある場合、上の環境変数の設定でソートさせると、

| | | | | |
|---|---|---|---|---|
| test | doc | 6802 | 93-01-19 | 1：44：02 |
| test | tex | 6802 | 93-01-19 | 1：44：02 |
| test | r | 6802 | 93-01-19 | 1：44：02 |
| tsort | s | 29257 | 92-02-14 | 3：03：46 |
| tsort | doc | 8670 | 92-02-14 | 3：46：44 |
| tsort | tex | 10031 | 92-02-14 | 4：43：38 |
| tsort | r | 4516 | 92-02-14 | 3：04：04 |

のように、ソース、ドキュメント、実行プログラムの順になります。

もし、拡張子が優先されるオプション“-e”がコマンドライン上で設定されていた場合は、

| | | | | |
|---|---|---|---|---|
| tsort | s | 29257 | 92-02-14 | 3：03：46 |
| test | doc | 6802 | 93-01-19 | 1：44：02 |
| tsort | doc | 8670 | 92-02-14 | 3：46：44 |
| test | tex | 6802 | 93-01-19 | 1：44：02 |
| tsort | tex | 10031 | 92-02-14 | 4：43：38 |
| tsort | r | 4516 | 92-02-14 | 3：04：04 |
| test | r | 6802 | 93-01-19 | 1：44：02 |

となります。tsortextで指定された拡張子を持っていないファイルがある場合は、そのファイルが最後になります。

**●優先順位**

　複数のオプションを同時に指定した場合、ソートの優先順位は、

**タイムスタンプ→ファイルサイズ→ファイルネーム**

です。これは、タイムスタンプが同じだったらファイルサイズで、ファイルサイズも同じだったらファイルネームでソートする、ということです。

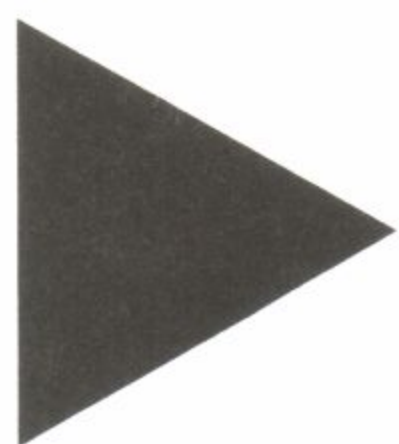
**●オプション -n について**

　このオプションスイッチは何のためにあるのかというと、作者によると、「他人に見せびらかすため」だそうです。オプションスイッチ“-n”をつけると、

```
cmbx9.pk
cmbx09.pk
```

という、２つのファイルは区別できません。しかし、このあと、友人の目の前で“-n”をとってソートさせて見せるわけです。

```
cmbx09.pk
cmbx9.pk
```

となるわけです。そうすると、友人は驚くかもしれません!?

# dedit.x

## X68000ユーザ必携の多機能ディスクエディタ

**【概要】**

フロッピーディスクなどの媒体は、Fig.1のようにトラック、セクタ、という単位に分割されており、どのトラックの、どのセクタといった具合に場所が指定されてデータの書き込みや読み出しが行われます。しかし、我々ユーザ側は、ディスクの中に入っているデータを読み書きする場合でも、トラックやセクタといった場所を指定する必要はありません。単に読み書きしたいデータのファイル名さえ覚えていればよいのです。

ファイル名に対し、それが実際にディスク上のどこに格納されているのかを管理してくれるのが、ディスクオペレーティングシステム（X68000の場合は、Human68kのことです。以下、ディスクオペレーティングシステムのことをHuman68kと呼びます）の仕事なのです。

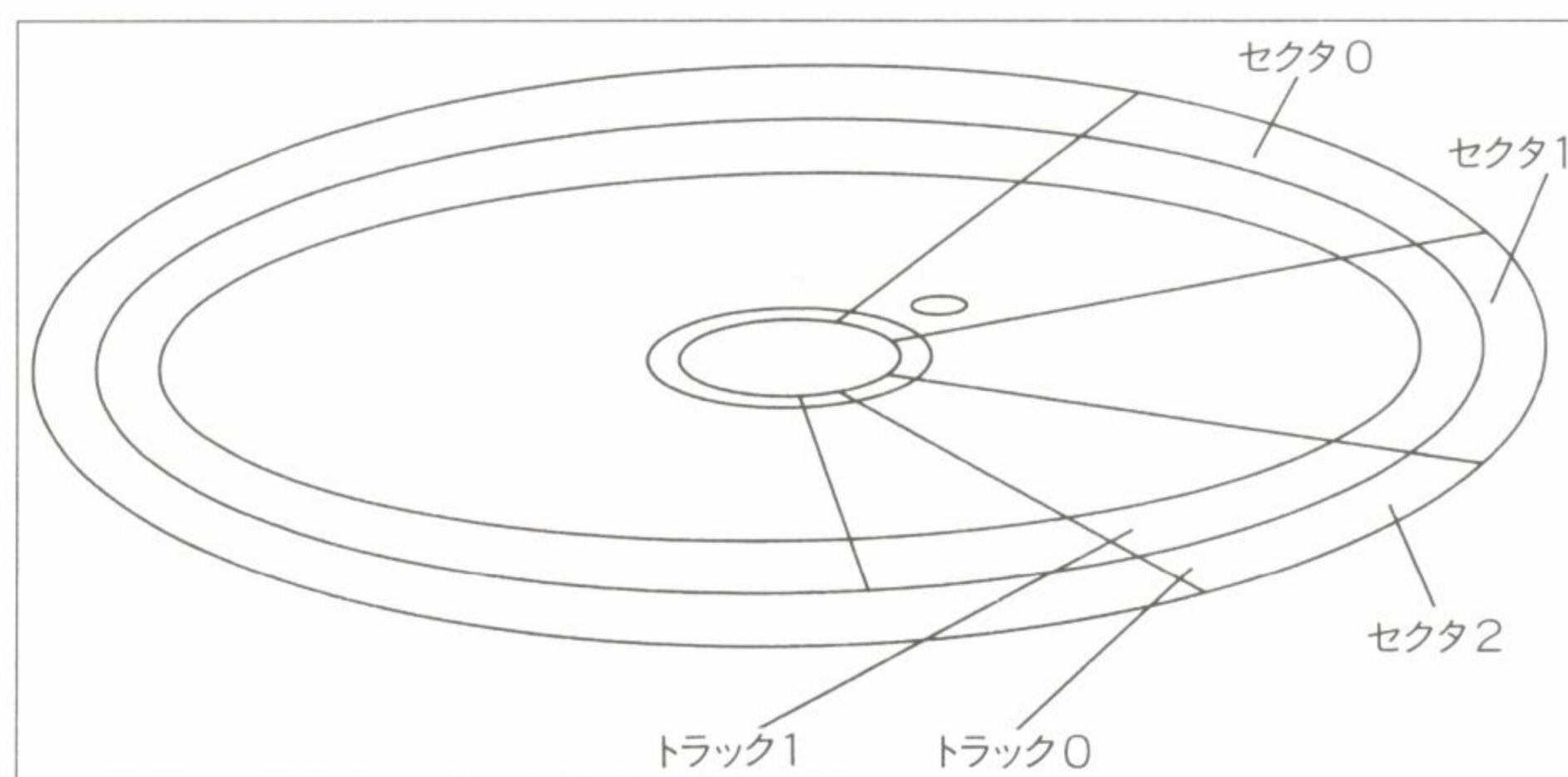
ところが、なんらかの理由で、この大事なディスクの管理領域が壊れてしまったとしたらどうでしょう。また、うっかり大事なファイルをデリートしてしまったとしたら、Human68kは、データが読み出せない、もしくは存在しないと答えるでしょう。

たとえ、ディスク上の実際のデータは失われていなくても、その管理情報が失われただけでHuman68kはデータを読み出すことができなくなるのです。このようなときには、すぐにセクタ単位でデータを読み書きするdeditが威力を発揮します。

この他にもdeditは実に多彩な機能を持っています。主なところを挙げてみましょう。

(1) ED.Xなどのプログラムや文書用のエディタでは編集できないバイナリファイルを16進ダンプの形で編集する機能



Fig.1　ディスク上のトラックやセクタを示す概略図

(2) ディレクトリの中身をファイル名順にソートしたり、ファイルを削除してできたディレクトリの隙間を詰めたりするディレクトリ編集機能
(3) メモリ内容を16進ダンプの形で編集する機能
(4) 特殊フォーマットのフロッピーディスクを読み書きする機能

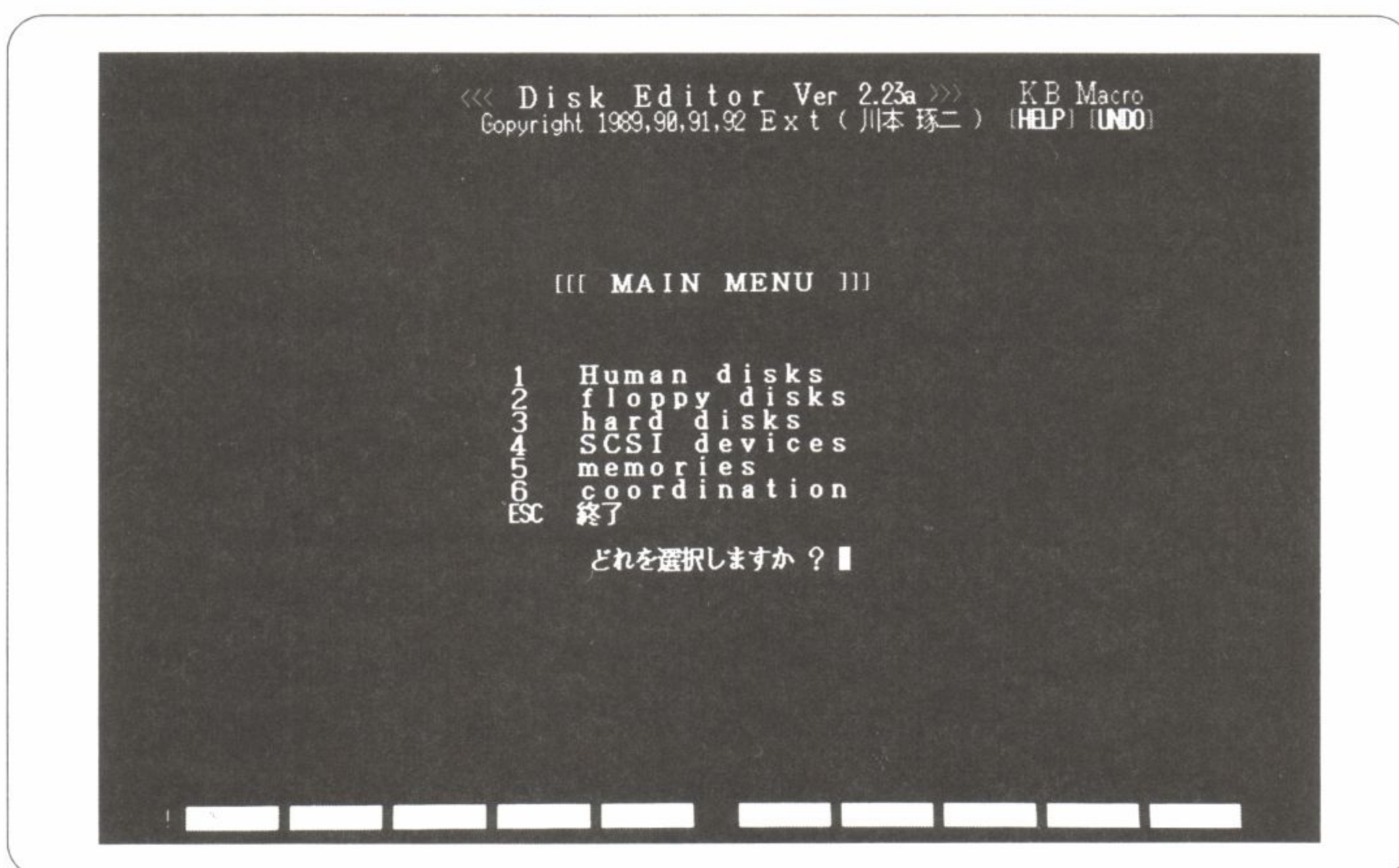
など、セクタ単位のディスクエディタにとどまらない機能を持っています。

【作者】……………………Ext(川本琢二)　NIFTY-Serve　NAH00720
E-Mail　kawamoto@miln.mei.co.jp

【使用する前に】………　deditは、ディスクエディタという性格上、多彩な機能をフルに使いこなすには、トラック、セクタ、ディレクトリ、ファイルアロケーションテーブル(File Allocation Table=FAT。後述)等のディスクに関する専門的な知識がある程度必要となります。deditを不用意に使うとファイルを破壊する可能性もありますので、使用には十分な注意が必要です。

【主な使用法】…………　deditをオプションなしで立ち上げると、Pht.1のような画面となります。これがdeditのメインメニューです。deditの各機能は、おのおののメニューから選択することができます。

Pht.1　deditのメインメニュー

```
<<< Disk Editor Ver 2.23a >>>   KB Macro
Copyright 1989,90,91,92 Ext ( 川本 琢二 ) [HELP] [UNDO]

[[[ MAIN MENU ]]]

1   Human disks
2   floppy disks
3   hard disks
4   SCSI devices
5   memories
6   coordination
ESC 終了

どれを選択しますか ?
```

また、起動時にオプションスイッチを指定することで、メニューを省略して直接、各機

能を呼び出すこともできます。

【書式】……………………dedit［オプションスイッチ］［ファイル名］

注：ファイル名を指定した場合は、ファイルエディット機能で起動する。

【オプションスイッチ】

| オプションスイッチ | 内　容 |
| --- | --- |
| -timer＝タイマー値 | フロッピーディスクの特殊フォーマットを識別する[注] |
| -macro＝マクロファイル名 | あらかじめキーボード操作を記録したマクロファイルを指定し、一連の動作を自動実行させる |

注：nonstandard track edit（後述）においてフロッピーディスクが1回転する時間を計測するためのタイマー値を設定する。デフォルトでは10000となっている。SX-WINDOWなどから起動する場合のように、内部タイマーにずれが生じるソフトを使用するときは、このタイマー値を変更する必要がある。詳しくは、ドキュメントを参照のこと。

以下のオプションは、メニューを省略してdeditの各機能を直接呼び出すためのものです。

| オプションスイッチ | 内　容 |
| --- | --- |
| -H | Human disksモードのメニュー画面で起動する |
| -Hs | sector editモードの編集画面で起動する |
| -Hf | file editモードの編集画面で起動する |
| -Hd | directory editモードの編集画面で起動する |
| -Hu | undelモードの編集画面で起動する |
| -f | floppy disksモードのメニュー画面で起動する |
| -ft | nonstandard track editモードの編集画面で起動する |
| -fr | read diagnosticモードの編集画面で起動する |
| -fv | visualモードの編集画面で起動する |
| -h | hard disksモードのメニュー画面で起動する |
| -s | scsi devicesモードのメニュー画面で起動する |
| -m | memoriesモードの編集画面で起動する |

【使用法】……………………

### 1．Human68kディスクのエディットをする場合

メインメニューから「1 Human disks」を選びます。

X68000には、フロッピーディスク、ハードディスク、RAMディスクなどのさまざまなディスクがありますが、Human68kからは、論理的なセクタ番号により系統的にアクセスされます。物理的なセクタサイズやトラックなどのディスクごとの違いは、デバイスドライバと呼ばれる個別のプログラムにより吸収されています。

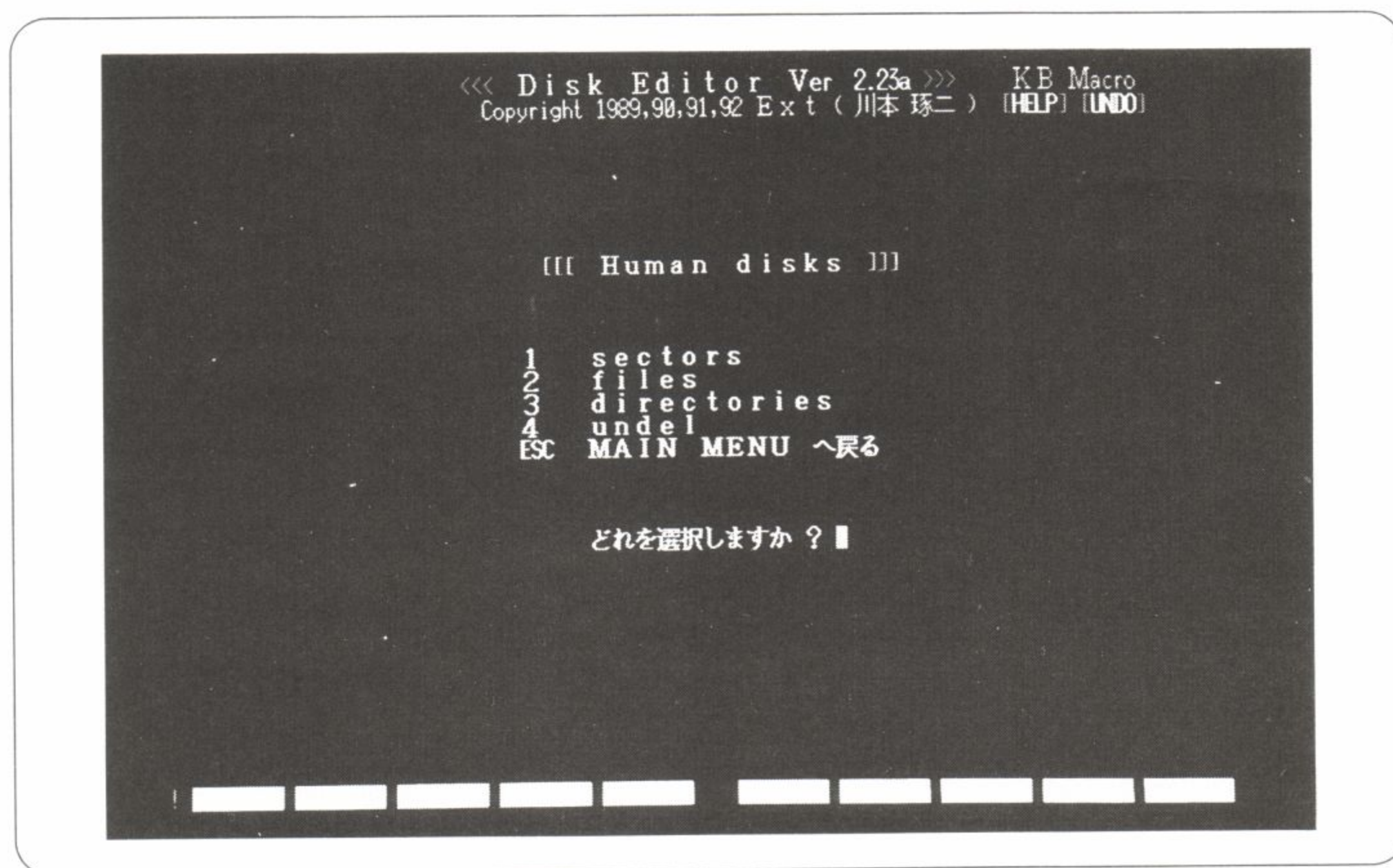
deditのHuman diskのエディット機能は、この論理セクタ単位でエディットを行うものです。どのディスク上のデータをエディットするかという指定は、Human68kが割り付けるドライブ名（A:～Z:）で行います。

なお、Human68kでは、物理セクタ、論理セクタの他に、クラスタという単位も使われています。これは、ファイルの書き込まれたセクタの順番を示すファイルアロケーション

テーブルで使われている管理番号です。deditでは、削除したファイルの復活を行うundel機能の中でクラスタ番号を使用しています。

### 1-1 Human diskのセクタエディット

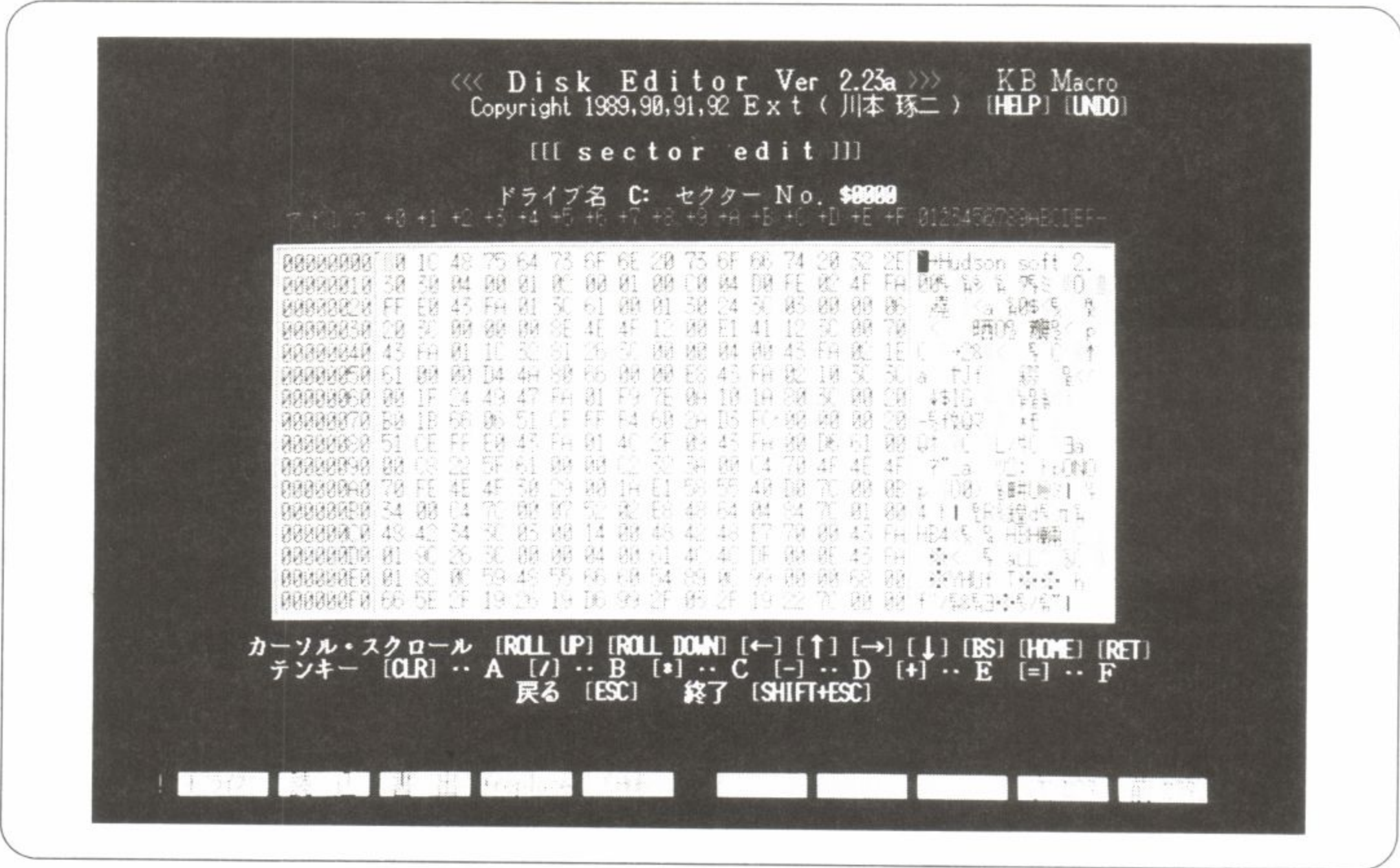
メインメニューから「1 Human disks」を選ぶと、Pht.2のようなサブメニューが表示されます。

Pht.2 Human diskをエディットする場合のサブメニュー

このサブメニューから「1　sectors」を選ぶとセクタエディット機能となり、Pht.3のようなセクタエディット画面となります。deditをオプションスイッチ"-Hs"をつけて起動した場合は、メニューを飛ばして直接このセクタエディット画面になります。

画面中央の大きな枠内が編集中のセクタの内容です。左側に16進数のダンプ形式で、右側にはそのデータがシフトJISコードで表示されます。画面上部にはセクタ番号などの情報が、画面下部にはファンクションキーおよび使用可能なキーが表示されています。

カーソルキーやスクロールキーを使うことにより、16進ダンプ表示されたセクタ内のデータ上をカーソルが移動します。変更したいデータのところにカーソルをあわせ、直接データを入力することで、データを変更できます。ただし、実際には、[F3]の(書き出し)を行わない限り、変更したセクタの内容はディスク上には書き込まれません。セクタ単位で「書き出し」を行うのは、deditの他のエディット機能にも共通した操作です。これは一見面倒にも思えますが、deditでは画面上で簡単にデータの書き換えができてしまいますので、不用意に変えてしまったものが、そのままディスク上に書き込まれてしまわないようにす

るためにはしかたがないでしょう。

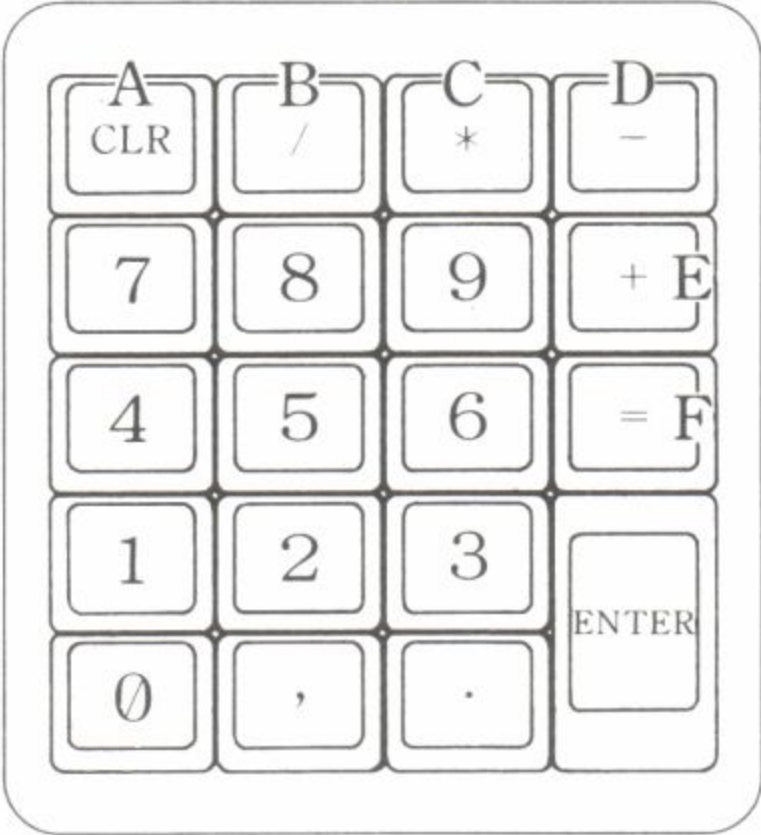
Pht.3 Human diskのsector edit画面

逆に、画面上で変更したのに「書き出し」を忘れていた場合は、他のセクタへ移動しようとしたときに、「バッファの内容書き換えが有ります。本当にセクタNoを変更しますか？(y/n)」と確認してきますので、安心です。

これは、deditを終了しようとするときも同様で、「バッファの内容書き換えが有ります。本当に終了しますか？(y/n)」と確認してきます。

セクタエディットでは以下のキーが使用できます。

| キー | 表示 | 内容 |
|---|---|---|
| [F1] | ドライブ | エディット対象のドライブを変更する |
| [F2] | 読込 | 目的セクタの内容を読み込む |
| [F3] | 書出 | 編集結果をセクタに書き出す |
| [F4] | replace | 通常は、変更しようとして入力したデータは、変更前のデータに置き換わる。しかし、[F4]キーにより、単なるデータの置き換えでない変更方法注1を指定できる |
| [F5] | CHAR | 右側のカーソルが点滅し、キーボードから入力した文字がそのままデータとなる。FEPにより漢字を入力した場合も、そのままシフトJISのデータとして入力される<br>コントロールコードを入力する場合は、[TAB] キーを押してから英字を入力する<br>なお、通常の16進数入力に戻すのは、[F4](HEX)キーを押す |
| [F12]<br>[F13] | ロード<br>セーブ | ロードとセーブの機能は、現在エディット中のセクタの内容をHuman68kのファイルに読み書きするもの。あるセクタの内容をHuman68kのファイルとしてセーブしておき、他のセクタにロードすることによりセクタの内容をコピーすることができる |
| [F14] | サーチ | 指定した16進数のデータ列もしくは文字列を検索する機能注2 |
| [F15] | 次サーチ | [F14]で指定した文字列で再検索する |

| | | |
|---|---|---|
| [OPT.1]+<br>[ROLL UP]<br>[F9] | 次セクタ | 次のセクタに移動 |
| [OPT.1]+<br>[ROLL DOWN]<br>[F10] | 前セクタ | 前のセクタに移動 |
| [SHIFT]+<br>[OPT.1]+<br>[ROLL UP]<br>[F19] | 頭セクタ | 先頭セクタに移動 |
| [SHIFT]+<br>[OPT.1]+<br>[ROLL DOWN]<br>[F20] | セクタNo | セクタ番号を入力して、その番号のセクタに移る |
| テンキー | | テンキーから16進数が入力できるよう、Fig.2のように、[A][B][C][D][E][F]が割り当てられている |
| [ESC] | | セクタエディットを終了し、「1 Human disks」のメニュー画面に戻る |
| [SHIFT]+<br>[ESC] | | deditを終了する<br>deditでは、基本的に[ESC]キーで1つ前の画面に戻り、[SHIFT]+[ESC]で一気に終了する |

**Fig.2** セクタエディットでのテンキーの状態

注：

1）[F4]キーを押すたびに、ファンクションキーの表示は"replace"→"and"→"or"→"xor"→"replace"と変化する。and、or、xorでは変更前のデータと入力したデータのビットごとの演算が行われる。

例)変更前のデータが16進数で4A(ビットで01001010)で、入力したデータが6C(ビットで01101100)の場合、

| [F4]の表示 | 変更後のデータ |
|---|---|
| replace | 6C |
| and | 4A and 6C → 48 |
| or | 4A or 6C → 6E |
| xor | 4A xor 6C → 26 |

となる。

2）[F14]キーを押すと、

検索文字列([hex] / s[string]) ?

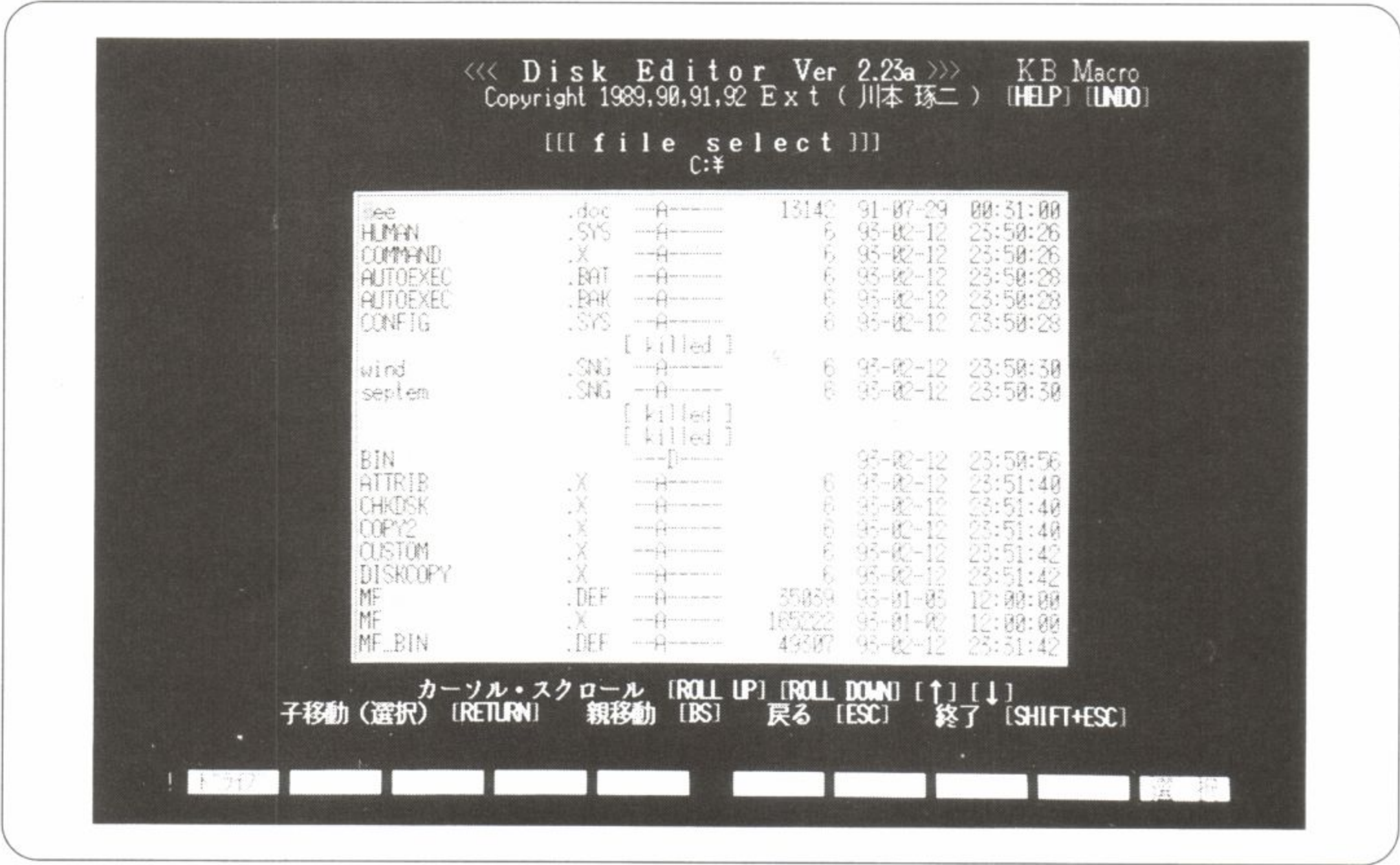
と表示されるので、16進データの場合は“0A0B0C”という具合に連続して、文字列の場合は最初に“s”をつけて“s検索対象文字列”といった具合に入力する。
　検索文字列を指定すると、

方向(f/b)?

と表示される。これは、現在カーソルのある位置から後ろのデータの中から検索するか(f)、前のデータから検索するか(b)を指定するものである。
　以上の指定が完了すると、検索が始まる。検索を中断する場合は[ESC] キーを押す。

### 1-2 Human diskのファイルエディット

「1 Human disks」のサブメニューから「2 files」を選ぶとファイルエディット機能となり、Pht.4のようなファイル選択画面となります。また、deditをオプションスイッチ“-Hf”をつけて起動した場合は、直接このファイル選択画面となります。“dedit ファイル名”というように、ファイル名を指定して起動した場合は、ファイル選択を飛ばしてファイル編集となります。

**Pht.4** filesモードのファイル選択画面

#### ●ファイル選択

カーソルキーやスクロールキーでカーソルを移動し、[CR]キーを押すことによって、そのファイルのエディット画面となります。

ディレクトリの上で[CR]キーを押した場合は、サブディレクトリに移動し、そこでのファイル選択画面となります。また、親ディレクトリへの移動は[BS]キーまたは[ESC]キーで行います。

ディレクトリの上で[CR]キーを押すかわりに[F10](選択)キーを押すと、そのディレクトリを仮想的にファイルとみなしてエディットできるようになります。この場合、ファイル名やファイルの更新日付といった、本来そのディレクトリに収められているファイルに対する管理情報も、単なる16進ダンプのデータとして扱われることになります。

単にファイル名や日付などを変更するといった目的には、後述するディレクトリエディットのほうが便利ですが、ファイルエディットを使って仮想的なファイルとして編集することで、セーブ、ロードなどの機能を使うことができます。

### ●ファイルエディット画面

ファイルの選択をすると、ファイルのエディット画面となります。deditのファイルエディットは、ファイル全体に対しての編集ではなく、そのファイルが存在する個々のセクタ単位で編集していきます。セクタエディットとほぼ同じ操作で、セクタ単位で変更結果を書き出さないといけないことも同じです。

セクタエディットと異なるのは、以下のとおりです。

| キー | 表示 | 内　容 |
|---|---|---|
| [F8] | アドレス | ファイル先頭を0番地として、ファイル内の指定のアドレスに移動する |
| [F9] | 次セクタ | ファイルの次のセクタに移動する |
| [F10] | 前セクタ | ファイルの前のセクタに移動する |

セクタエディットでは論理セクタ順に移動しましたが、ファイルは必ずしも論理セクタ順になっているわけではありません。しかし、ファイルエディットでは、ディスク上でファイルが格納されている順にセクタ移動します。

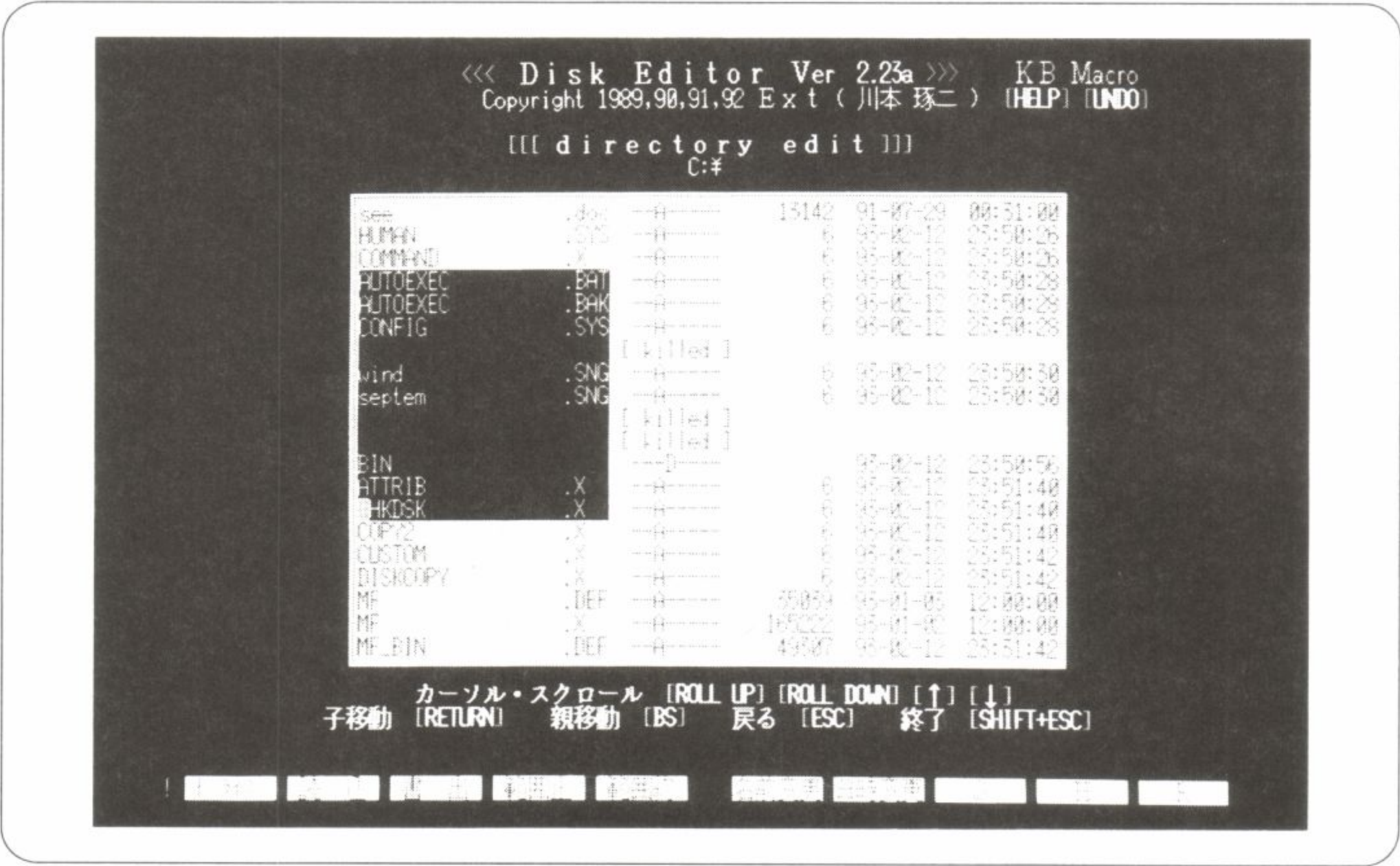
## 1-3 Human diskのディレクトリエディット

「1 Human disks」のサブメニューから「3 directories」を選ぶとディレクトリエディット機能となり、Pht.5のようなディレクトリ編集画面となります。また、deditをオプションスイッチ“-Hd”つきで起動した場合は、直接この画面となります。

ディレクトリエディットとは、ファイル名、属性、日時などのファイルの管理情報を専門に編集する機能です。また、ディレクトリ上でのファイル名の位置の並べ替えなどの機能もあり、これによりDIRコマンドなどで表示される順番を好きなように変えることができます。

ディレクトリエディットでも、セクタエディットなどと同様、画面上でいくら変更してもユーザが実際に書き出しを指定するまでは編集結果はディスクに書き込まれませんので、

いろいろ変更しながら試してみることができます。

Pht.5　ディレクトリ編集画面。範囲の指定をしているところ

ディレクトリの編集には、以下のキーを使用します。

| キー | 表示 | 内　容 |
|---|---|---|
| [F2] | 読込 | ディレクトリデータの再読み込み |
| [F3] | 書出 | 編集したディレクトリデータをディスクに書き込む |
| [F4] | 範囲始 | 並べ替えなどの機能を実行するときの範囲を指定する<br>範囲の始めで[F4]キーを押す。範囲選択中はファンクションキーの表示が、[F4](範囲止)、[F5](範囲終)に変わる。カーソルを動かすと範囲が反転表示されるので、選択したい範囲の最後で[F5]キーを押す |
| [F5] | 並べ替 | [F4]で選択したファイルのディレクトリ上での位置を移動する。カーソルを移動させたい位置に持っていき、[F5]キーを押すと、その位置の後ろに選択したファイルが移動する |
| [F6] | 名前変更 | カーソル位置のファイルの名前を変更する。ディレクトリ名やボリューム名も変更できる |
| [F7] | 日時変更 | カーソル位置のファイルの日時を変更する。[F4]キーで範囲指定してあった場合は、その中のファイルすべての日時が変更される<br>新しい日付と新しい時刻をそれぞれ聞いてくるので入力する<br>なお、何も入力せずに[CR] キーだけを押すと、現在の日付や時刻にしたり、空欄にしたりするメニューが表示される |
| [F8] | S | カーソル位置のファイルの「システム属性」を変更する。[F4]キーで範囲指定してある場合は、その中のファイルすべてのシステム属性が変更される<br>ただし、この変更は、もともとシステム属性がついていたファイルに対して、そのシステム属性をオン／オフするもので、ディレクトリ名やボリューム名は対象外である |
| [F9] | H | カーソル位置のファイルの「ヒドゥン属性（不可視属性)」を変更する。[F4]キーで範囲指定してある場合は、その中のファイルすべてのヒドゥン属性が変更される |

| | | |
|---|---|---|
| [F10] | R | カーソル位置のファイルの「リードオンリー属性」を変更する。[F4]キーで範囲指定してある場合は、その中のファイルすべてのリードオンリー属性が変更される |
| [F11] | ASCソート | ディレクトリエントリをファイル名のASCIIコード順に並べ替える<br>[F4]キーで範囲指定してある場合は、範囲内のファイルについてのみソートされるが、範囲指定されていない場合は、そのディレクトリ内のすべてのファイルが並べ替えられる<br>なお、[F12][F13][F14]キーでソートの細かいモード指定が行える。Fig.3のソートとモードの関係を参照のこと |
| [F12] | Aa 別 | ASCIIソートにおいて、大文字、小文字の区別を指定する<br>[F12]キーを押すことで、「Aa 別」と「Aa 同」を切り替える<br>ファンクションキーの場所に現在のモードが表示される<br>　Aa 別：大文字、小文字を区別してソートする<br>　Aa 同：大文字、小文字を区別しないでソートする |
| [F13] | 記号 | ファイル名に数値を含む場合のソート方法を指定する<br>[F13]キーを押すことで切り替わる<br>　記号：数字もそのままのASCIIコード順でソートする<br>　数値：数字の場合は、桁数を考慮に入れてソートする |
| [F14] | 全体 | ソートの対象をファイル名全体にするかどうかを指定する<br>[F14]キーを押すことで切り替わる<br>　全体：ファイル名全体でソートする<br>　ノード：ファイル名の拡張子を除いた部分でソートする<br>　拡張子：ファイル名の拡張子だけでソートする |
| [F15] | Date ソート | ディレクトリエントリをファイルの更新日時順に並べ替える。[F4]キーで範囲指定してある場合は、範囲内のファイルについてのみソートされるが、範囲指定されていない場合は、そのディレクトリ内のすべてのファイルが並べ替えられる |
| [F16] | V-D-Aソート | ファイルの属性別に、ボリューム名、ディレクトリ、通常ファイルの順に並べ替える。[F4]キーで範囲指定してある場合は、範囲内のファイルについてのみソートされるが、範囲指定されていない場合は、そのディレクトリ内のすべてのファイルが並べ替えられる |
| [F17] | V-A-Dソート | [F16]とほぼ同じだが、ボリューム名、通常ファイル、ディレクトリの順に並べ替える |
| [F19] | 順序反転 | 並べ替えの順番を反転する |
| [F20] | 詰める | ディレクトリ上での隙間注を詰める。[F4]で範囲指定してある場合は、範囲内のファイルについてのみソートされるが、範囲指定されていない場合は、そのディレクトリ内のすべてのファイルが並べ替えられる |

注：DELコマンドなどのファイル削除は、ファイルが削除されたことを示す印がディレクトリ上のファイル名の位置に書き込まれることにより処理される。DIRコマンドなどでは、このファイル名は表示されなくなるが、ディレクトリエントリには隙間が残ってしまう。この状態で新しくファイルを作成すると、ディレクトリエントリの最後ではなく、この位置にできてしまうことになる。実害はないが、せっかくきれいに並べたディレクトリの並びが崩れるので、気持ちのよいものではない。

**Fig.3** ソートとモードの関係

**Fig.3-1** デフォルトのASCIIコード順ソート

| ソート前 | ソート後 |
|---|---|
| ZMUS10.dat | ABC.ZZZ |
| ZMUS2.dat | ZMUS10.dat |
| abc.def | ZMUS2.dat |
| ABC.ZZZ | abc.def |

**Fig.3-2** 「Aa 同」のASCIIコード順ソート

| ソート前 | ソート後 |
|---|---|
| ZMUS10.dat | abc1.def |
| ZMUS2.dat | ABC2.ZZZ |
| abc1.def | ZMUS10.dat |
| ABC2.ZZZ | ZMUS2.dat |

Fig.3-3 「数値」のASCIIコード順ソート

| ソート前 | ソート後 |
|---|---|
| ZMUS10.dat | ABC2.ZZZ |
| ZMUS2.dat | ZMUS2.dat |
| abc1.def | ZMUS10.dat |
| ABC2.ZZZ | abc1.def |

Fig.3-4 「拡張子」のASCIIコード順ソート

| ソート前 | ソート後 |
|---|---|
| ZMUS10.dat | ABC2.ZZZ |
| ZMUS2.dat | ZMUS10.dat |
| abc1.def | ZMUS2.dat |
| ABC2.ZZZ | abc1.def |

### 1-4 Human diskのundel

undel機能は、誤ってデリートしてしまったファイルを復活させるためのものです。DELコマンドも含め、Human68kでのファイル削除は、普通、ファイル内容そのものを消すのではなく、ファイルを管理している部分を消すだけです。したがって、削除されてしまったファイルでも、ディスク上に残っているデータを直接読み出すことで、ファイルを復活させることが可能です。

ただし、なんでも復活できるわけではなく、削除後に他のファイルを書き込んだりしてディスク上に残っているデータが上書きされてしまっている場合などは不可能です。また、ファイルが複数のクラスタにまたがっていた場合、管理データの一部が消えているため、簡単には修復できない場合があります。どういう方法でundelが行われているのかを理解すれば、どういうときに使えて、どんな制限があるかがわかってきます。詳細は、コラムを参照してください。

---

**COLUMN**

#### Human68kのファイル管理とファイル削除の仕組み

undel機能を使いこなすためにも、ある程度Human68kでのファイル管理の仕組みを理解しておく必要がありますので、ここで簡単に説明しておきましょう。

Human68kでは、ディレクトリとファイルアロケーションテーブル（以降、FATと略す）によりファイルを管理しています。

ディレクトリには、Fig.4-1のように、ファイルごとにファイルの名前、ファイルサイズ、作成日付、リードオンリー（読み出し専用）やヒドゥン（不可視）などの属性とともに、ファイルの格納されている先頭のクラスタ番号が記録されています。

FATは、Fig.4-2のように、テーブル上の1つ1つのデータが、それぞれディスク上の1つのクラスタに対応しており、どのクラスタが空いているかということと、複数のクラスタにまたがっているファイルがどのクラスタの順番に格納されているかを示す情報が書き込まれています。

たとえば、Fig.4-1でファイル“abc.def”は、ディレクトリの情報から最初の部分がクラスタの5番にあることがわかります。そして、Fig.4-2からクラスタの5番目の次はクラスタの7番であることがわかります。こうやってFATを辿っていくと、次の7番目には、これがファイルの最後であることを示す特殊な番号が書き込まれています。このように、FAT上に書かれたクラスタ番号の並びを「クラスタチェイン」と呼びます。

Human68kがファイルをアクセスする場合も、このように、ファイル名から実際のデ

**Fig.4-1** ディレクトリとFAT(ディレクトリの内容)

| | +0 | +1 | +2 | +3 | +4 | +5 | +6 | +7 | +8 | +9 | +A | +B | +C | +D | +E | +F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0020 | 61 | 62 | 63 | 20 | 20 | 20 | 20 | 20 | 64 | 65 | 66 | 20 | 00 | 00 | 00 | 00 |
| 0030 | 00 | 00 | 00 | 00 | 00 | 00 | 4A | 7F | 30 | 1A | 05 | 00 | D2 | 04 | 00 | 00 |
| 0040 | 64 | 65 | 66 | 20 | 20 | 20 | 20 | 20 | 67 | 68 | 69 | 20 | 00 | 00 | 00 | 00 |
| 0050 | 00 | 00 | 00 | 00 | 00 | 00 | 35 | 80 | 30 | 1A | 06 | 00 | 00 | 03 | 00 | 00 |

ファイル名
abc. def
最終変更月日
開始セクタ番号
アトリビュート
ファイル名
def. ghi
最終変更月日
開始セクタ番号
ファイルサイズ

**Fig.4-2** ディレクトリとFAT(FATの内容)

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0000 | FFE | FFF | 003 | 004 | 終 | 007 | 終 | 終 | 空 | 空 | 空 | 空 |
| 0010 | 空 | 空 | 空 | 空 | 空 | 空 | 空 | 空 | 空 | 空 | 空 | 空 |

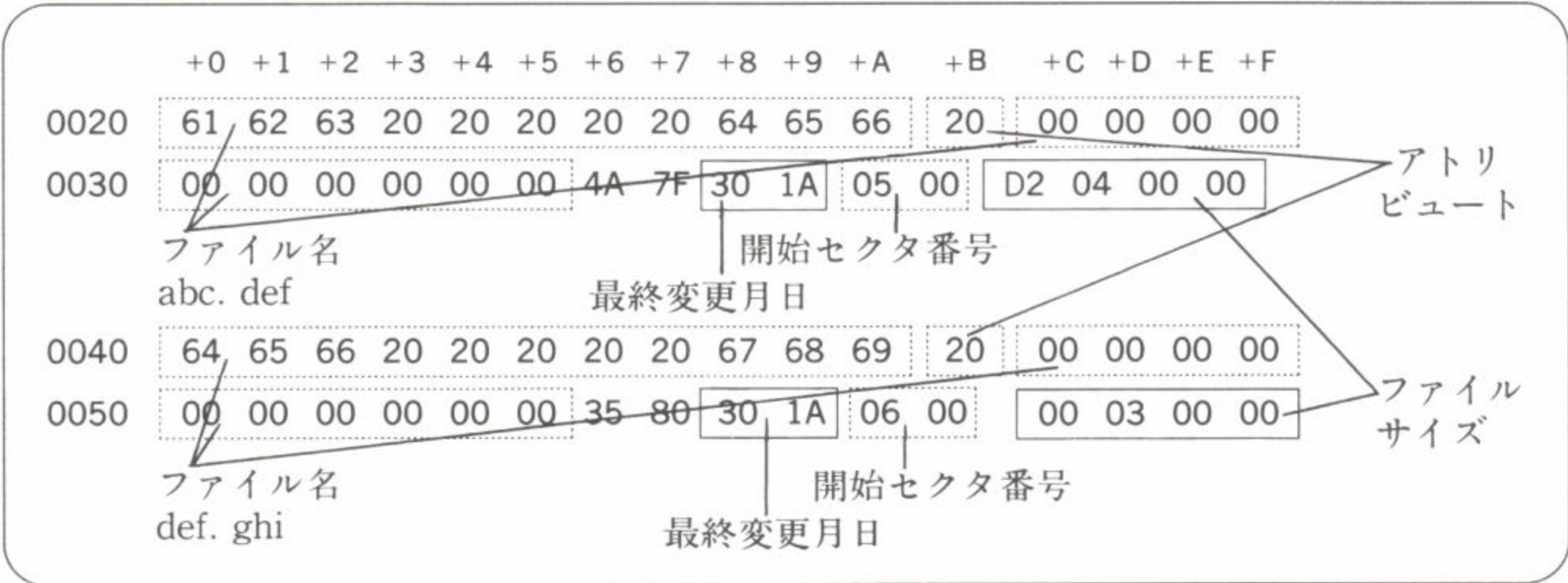
次のクラスタを指す
ファイルのabc. defのディレクトリエントリから指される

ータの存在するクラスタを辿っているのです。

Fig.5にファイル“abc.def”を削除したあとのディレクトリとFATの状態を示します。Fig.5-1でディレクトリ上のファイル名“abc.def”の部分は、先頭の“a”のあったところに削除マークとして16進数で“E5”が書き込まれているのがわかります。以後、Human68kはディレクトリ上のこの部分は「空」と思って処理し、DIRコマンドでも表示されなくなりますが、新しいファイルを書き込んだりしない限り、ファイル名の残りの部分やファイルサイズなどのファイル名の先頭以外のデータは残っています。

**Fig.5-1** ファイルを削除したあとのディレクトリとFAT(ディレクトリの内容)

| | +0 | +1 | +2 | +3 | +4 | +5 | +6 | +7 | +8 | +9 | +A | +B | +C | +D | +E | +F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0020 | E5 | 62 | 63 | 20 | 20 | 20 | 20 | 20 | 64 | 65 | 66 | 20 | 00 | 00 | 00 | 00 |
| 0030 | 00 | 00 | 00 | 00 | 00 | 00 | 4A | 7F | 30 | 1A | 05 | 00 | D2 | 04 | 00 | 00 |
| 0040 | 64 | 65 | 66 | 20 | 20 | 20 | 20 | 20 | 67 | 68 | 69 | 20 | 00 | 00 | 00 | 00 |
| 0050 | 00 | 00 | 00 | 00 | 00 | 00 | 35 | 80 | 30 | 1A | 06 | 00 | 00 | 03 | 00 | 00 |

ファイル名の先頭にE5が書き込まれる

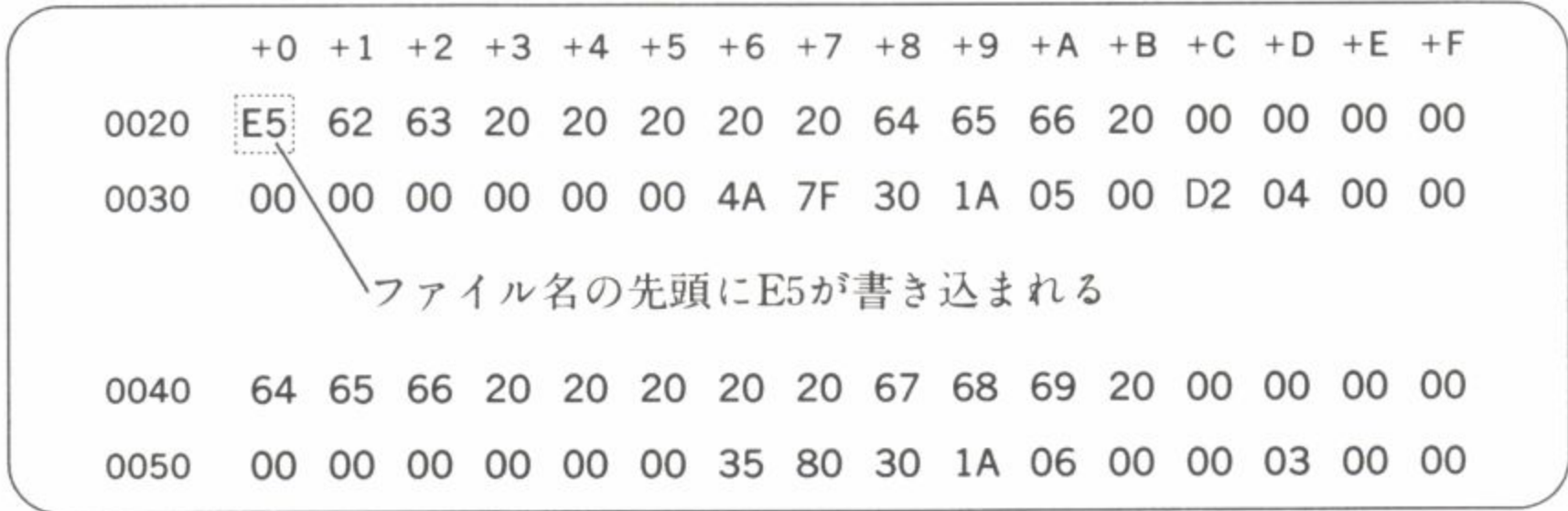



**Fig.5-2** ファイルを削除したあとのディレクトリとFAT(FATの内容)

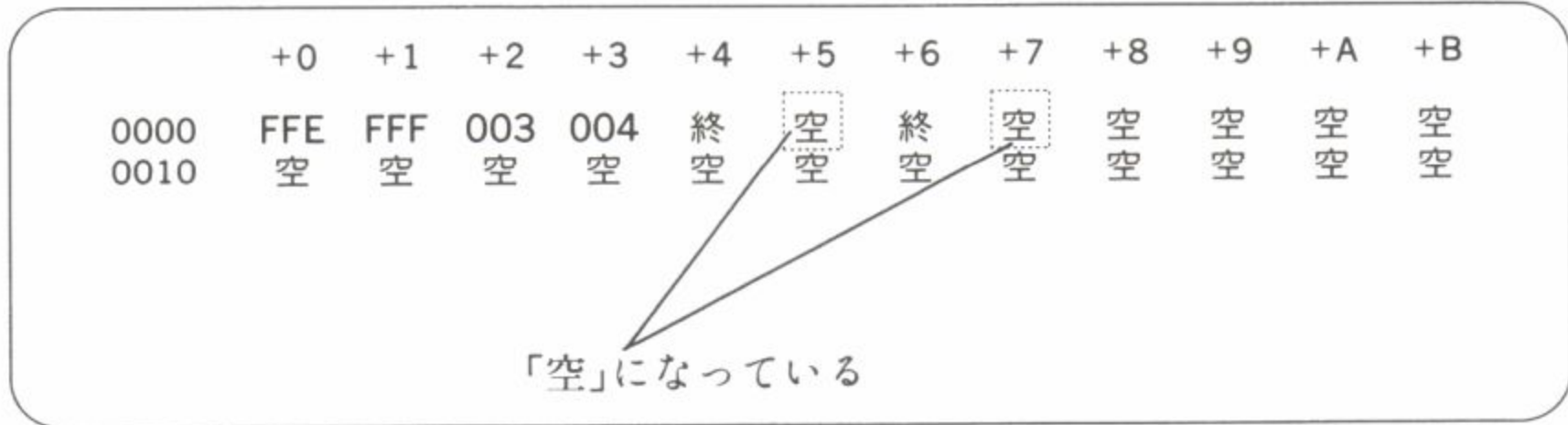
| | +0 | +1 | +2 | +3 | +4 | +5 | +6 | +7 | +8 | +9 | +A | +B |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0000 | FFE | FFF | 003 | 004 | 終 | 空 | 終 | 空 | 空 | 空 | 空 | 空 |
| 0010 | 空 | 空 | 空 | 空 | 空 | 空 | 空 | 空 | 空 | 空 | 空 | 空 |

「空」になっている

Fig.5-2では、FATファイルのあったクラスタの5番、7番は「空」になっていることを示す特殊な番号が書き込まれていることがわかります。このため、クラスタを辿るこ

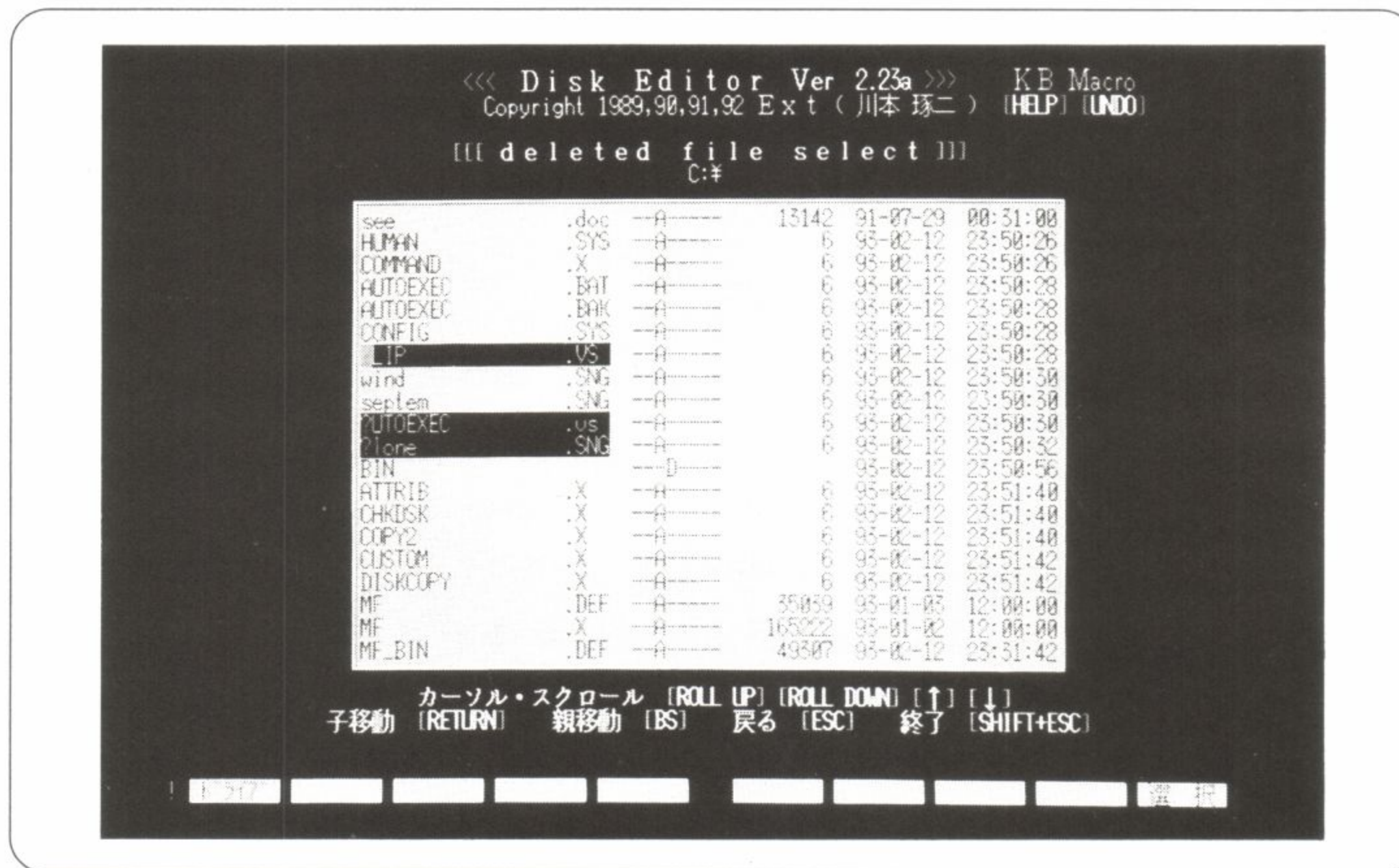
とはできなくなっています。
undelは、

1) ディレクトリの削除マークになってしまったファイル名の先頭文字を正しいファイル名に戻し、
2) 「空」とされてしまったFATを探して、おのおのに次のクラスタの番号を書き込んでいき、
3) 最後のクラスタにファイルの最後であることを示す特殊な番号を書き込む

という作業で、クラスタチェインを復活させ、FATに書き込む作業を行っているといえます。

### ●undelのファイル選択

undelの最初のステップは、削除されたファイルの選択です。Pht.6のようなファイル選択画面が表示されます。undelの対象となる削除されたファイルは、この一覧のうち、反転表示されたものだけです。削除されていないファイルは選択できません。

**Pht.6**　undelのファイル選択画面。削除されたファイルが反転表示されている

ここで、削除されたファイルのファイル名に注目してください。ファイルエディットなどのファイル選択画面では、削除されたファイルは[Killed]となって抜けていますが、undelのファイル選択画面では削除されたファイルのファイル名の先頭1文字だけが“?”になって表示されます。Human68kの削除処理により、ファイル名の先頭1文字目の部分に、削除マークの16進数で“E5”というデータが書き込まれるため、deditにはなんというファイル名だったかわからなくなっています。

ファイル名の残りの文字やファイルの更新日時からundelしたいファイルを推定して、カーソルをそこにあわせて[CR]キーで選択します。ファイルが選択されると、1文字目が何かを聞いてきます。これがわかるのは削除した本人ですので、1文字目が何だったかは人間が指定してやる必要があります。ただ、1文字目が多少違っても、違ったファイル名になるだけでファイルの中身には関係ありませんので、他のファイル名と同じにならないように適当な文字を指定してもかまいません。

**●FATダンプ画面**

undelするファイル名が確定したら、次はファイルが格納されていたクラスタを探していくことになります。

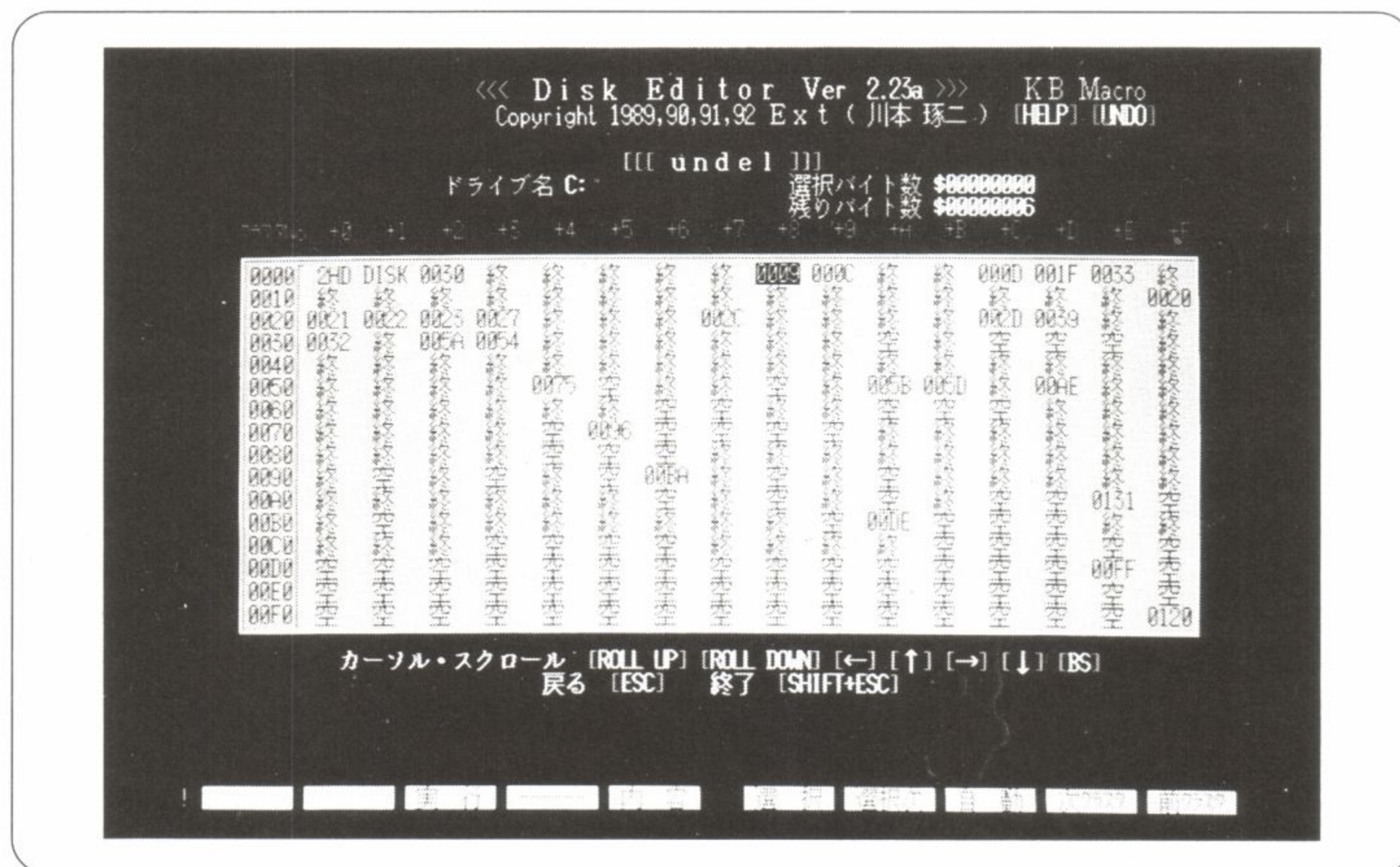
ディレクトリ上にはファイル名といっしょに先頭のクラスタの番号が入っており、この値自体は削除しても別のファイルを上書きしない限り残っているので、deditは選択されたファイルの先頭クラスタだけはわかります。

しかし、削除処理によりFAT内の次のクラスタの番号を指していた値が「空」を示す特殊な番号に上書きされてしまうため、ファイルが複数のクラスタにわたっている場合は、FATの情報を元に戻してHuman68kがクラスタを辿っていけるようにしてやらなければなりません。

この作業を行うのが、Pht.7のFATダンプ画面です。最初のカーソルのある位置は、先頭クラスタに対応する位置です。画面上で「空」と表示されている部分が、ファイルが削除されたか、もしくはもともと使われておらず空いていることを示すクラスタです。カーソルキーを使ってカーソルを「空」のクラスタにあわせ、[F6](選択)キーを押すことにより、選択したクラスタをクラスタチェインに組み込むことができます。

画面の上方には、選択したクラスタの合計バイト数と、ファイルサイズから逆算した残りのバイト数が示されています。削除されてしまったクラスタ数と同数のクラスタを選択すれば残りバイト数が0となりますので、どこまでクラスタを選択していけばいいかの目安になります。

Human68kでは、ファイルが作成される場合、普通はディスクの前のほうから順に空いているクラスタを使っていくので、削除されたファイルが1つなら、先頭のクラスタから順に「空」のクラスタを選択していくだけでOKの場合がほとんどです。このため、[F8]

**Pht.7** FATダンプ画面
番号(次クラスタを指す)か、"終"が書いてあるクラスタは使用中。"空"は削除されたか、未使用。ちなみに、エラーが出て使用不可になったクラスタは"死"と表示される。

(自動)キーにクラスタを自動的にファイルサイズ分選択する機能が用意されています。

すべてのクラスタを選択したら、[F3](実行)キーにより、ディレクトリおよびFATの情報が書き込まれ、削除されたファイルがアクセスできるようになります。

しかし、ファイルが複数削除された場合や、以前にファイルが削除されて、FAT上に多数の「空」のクラスタが散らばっている場合、また、複数のファイルの書き込みや削除を行うプログラム[注]で作られたファイルなどでは、「空」のクラスタが散らばってできますので、先頭クラスタから「空」を後ろに辿るだけでなく、前にも辿らなければならない場合もあります。

注：エディタで同時に複数のファイルを編集する場合などがこれにあたる。また、SX-WINDOWでは、疑似マルチタスク機能により、複数のプログラムを走らせることができるので、個々のプログラムが1つしかファイルの書き込みを行わなくても、全体から見ると複数のファイルの書き込み、削除が行われることになる。

こういう場合、「空」のクラスタの中身を見て判断するしかありません。FATダンプ画面上のカーソルがあるクラスタの内容は、[F5](内容)キーで内容を見ることができます。これで1つ1つ確認しながらクラスタを探していかなければならないでしょう。このため、ファイルの内容を見て、その内容の判断がつくようなテキストファイルの場合は可能ですが、実行形式のファイルであったり、複数ファイルを消してしまったときには、復活はあきらめたほうがいいかもしれません。

といっても、undelは一度はチャレンジしてみる価値があるでしょう。他にもundelができるツールはありますが、deditが最も強力です。deditでundelできないものは、他のツー

ルでもまず無理でしょう。

undelのFATダンプ画面では、以下のキーを使用します。

| キー | 表示 | 内　容 |
| --- | --- | --- |
| [F3] | 実行 | ディレクトリとFATの情報を書き込み、undelを実行する |
| [F4] | テーブル | [F5]の内容表示からFATダンプ画面に戻る |
| [F5] | 内容 | 内容表示画面にする。クラスタエディット画面とほぼ同じだが、データの変更はできない |
| [F6] | 選択 | 現在のクラスタを選択する |
| [F7] | 選択次 | 現在のクラスタを選択して、次のクラスタにカーソルを移動する。次のクラスタが他のファイルで使われている場合は、さらに、その次まで移動する |
| [F8] | 自動 | ファイルサイズに相当するセクタ数分の「空」クラスタを自動的に選択する |
| [OPT.1]+[ROLL UP]<br>[F9] | 次クラスタ | 次のクラスタにカーソルを移動する |
| [OPT.1]+[ROLL DOWN]<br>[F10] | 前クラスタ | 前のクラスタにカーソルを移動する |
| [F16] | 解除 | 選択したクラスタの一番最後のものを解除する |
| [F17] | 前解除 | 選択したクラスタの一番最後のものを解除し、そのクラスタにカーソルを移動する |
| [SHIFT]+[OPT.1]+[ROLL UP]<br>[F19] | 頭クラスタ | 先頭クラスタへカーソルを移動する<br>ただし、ディレクトリやFATなどの管理情報を格納する予約クラスタとして先頭から数クラスタが使われているため、[F19]で実際に移動するのは、通常のファイルが格納されるクラスタの先頭になる |
| [SHIFT]+[OPT.1]+[ROLL DOWN]<br>[F20] | クラスタNo | クラスタ番号を入力し、そこに移動する |

## 2．フロッピーディスクモードを使用する場合

オプションなしでdeditを起動し、メインメニューの「2　floppy disks」を選んで、次に出てくる4つのサブメニューから各機能を選ぶか、オプションスイッチ“-f”で直接、起動します。

フロッピーディスクモードについては専門知識が必要なので、本書では各機能の概要を説明するにとどめます。詳細については、deditのドキュメントファイルであるdedit.docを参照してください。

### ●nonstandard track editモード

「1 Human disks」では扱えなかった、OS-9などのHuman68k以外のフォーマットやゲームなどでの特殊なフォーマットのフロッピーディスクのエディットが行えます。

フロッピーディスク上のデータを、物理的なトラック、セクタ単位でエディットします。Human diskのセクタエディットと似たような操作ですが、画面上部にフロッピーディスクのタイプや記録密度などの各種のデータが表示されます。

**●read diagnosticモード**

「1　nonstandard track edit」と同様です。ただし、書き出しは行えません。

**●visualモード**

トラックの状態をグラフィック表示します。

**●cleaningモード**

クリーニングフロッピーを使ってフロッピーディスクドライブのヘッドをクリーニングするときに使用します。ディスク面全体をヘッドがまんべんなく通過します。この目的のためにのみ使用するのであれば、deditをオプションスイッチ"-fc"をつけて起動するほうがよいでしょう。

## 3．ハードディスクおよびSCSIデバイスモードを使用する場合

X68000では、SUPERより前の機種で採用されていた標準ハードディスクインタフェースと、SUPER以後およびSCSIインタフェースカードで使用できるハードディスクインタフェースが異なっています。deditでは、これらを区別しており、メインメニューからは、前者を「3　hard disks」、後者を「4　SCSI devices」として呼び出します。オプションスイッチで直接起動する場合でも、前者は"dedit -h"、後者は、"dedit -s"で行います。

機能は、どちらもディスク上のデータを、物理的なセクタ単位でエディットできます。操作は、Human diskのセクタエディットとほぼ同じです。

なお、このモードのセクタエディットでHuman68kで使用しているデータ領域をエディットすると不都合が起こる場合があるので[注]、OS-9などの他のOSの使用している領域や、IPL情報やパーティション情報などの、Human68kでも起動時以外は見ることのない領域にしか使用すべきではありません。

注：これらのセクタエディットは、Human68kの管理するクラスタより下位のレベルのIOCSプログラムで行われる。このため、Human68kの管理しているディスクでは、バッファリング機能やディスクキャッシュプログラムにより、バッファやキャッシュからディスクに書き戻されるとき、deditで変更したデータが元に戻ってしまうことが起こり得る。

**●パーティション領域の読み出し**

ハードディスクモードの例として、パィーティション情報を読み出してみましょう。

通常、ハードディスクなどの大容量記憶装置は、複数のパーティションに分割し、Human68k以外のOSの領域としたり、1つのOSの中でも複数のドライブとして使用することができます。このパーティション情報は、X68000のハードディスクの場合、物理的なセクタ番号の4番に書き込まれています。

Human68kでは、パーティションに分割された各領域の中で論理的なクラスタ番号を0番から割り振っているので、「1　Human disks」のセクタエディットでは物理的なセクタ

番号を指定して中身を見るということはできません。「3　hard disks」や「4　SCSI devices」を使うことで、物理的な番号で指定したセクタの内容を見ることができます。

40Mバイトのハードディスクを、

| | |
|---|---|
| Human68k | 領域2Mバイト |
| OS-9/68K | 領域2Mバイト |
| OS-9/68K | 領域17Mバイト |
| Human68k | 領域18Mバイト |

のようなパーティションに分割した場合、パーティション情報が書き込まれているセクタ４番の内容は、以下のようになります。

```
          +0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +A +B +C +D +E +F
00000000  58 36 38 4B 00 02 79 30 00 02 79 30 00 02 AC C0   X68K......
00000010  48 75 6D 61 6E 36 38 6B 00 00 00 21 00 00 1F A6   Human68k..
00000020  4F 53 2D 39 2F 36 38 4B 00 00 1F F8 00 00 1F F8   OS-9/68K..
00000030  4F 53 2D 39 2F 36 38 4B 02 00 3F F0 00 01 0F 38   OS-9/68K..
00000040  48 75 6D 61 6E 36 38 6B 02 01 5C 43 00 01 1C D6   Human68k..
```

各データの詳細については、「プログラマーズマニュアル」(シャープ㈱発行のXCコンパイラに付属) 等に説明が載っていますので、そちらを参照してください。大体の意味は、最初の16バイトにドライブ全体の情報が入り、次から16バイトごとに各パーティションの情報が入っています。

10番地からの８バイトに最初のパーティションのパーティション名であるHuman68kという文字列が入っているのがわかるでしょう。

同様にして、20番地からOS-9/68Kのパーティション情報が入っています。

ついでに、このパーティション情報を、deditの[F13](セーブ)によりフロッピーディスクなどの別のディスクにファイルとしてセーブしておくなり、メモしておくなりするとよいでしょう。もしハードディスクがクラッシュしてディスクが読み出せなくなったときは、まず、このパーティション情報を調べてみましょう。この情報が壊れていてディスクが読み出せなくなっているなら、セーブしておいたパーティション情報をロードするなり、メモに従ってエディットするなりしてハードディスクを復旧できます。

### 4. メモリエディットを使用する場合

Human68kは、システム起動時にメモリを初期化しないので、セーブする前にハングアップなどで失ったデータをメモリ上から取り出すことができます。メインメニューから「5

memories」を選択するか、“dedit　-m”で起動します。

メモリエディットは、メモリ上のデータを編集します。また、VRAMやI/Oポートに対しても直接書き込みすることができます。

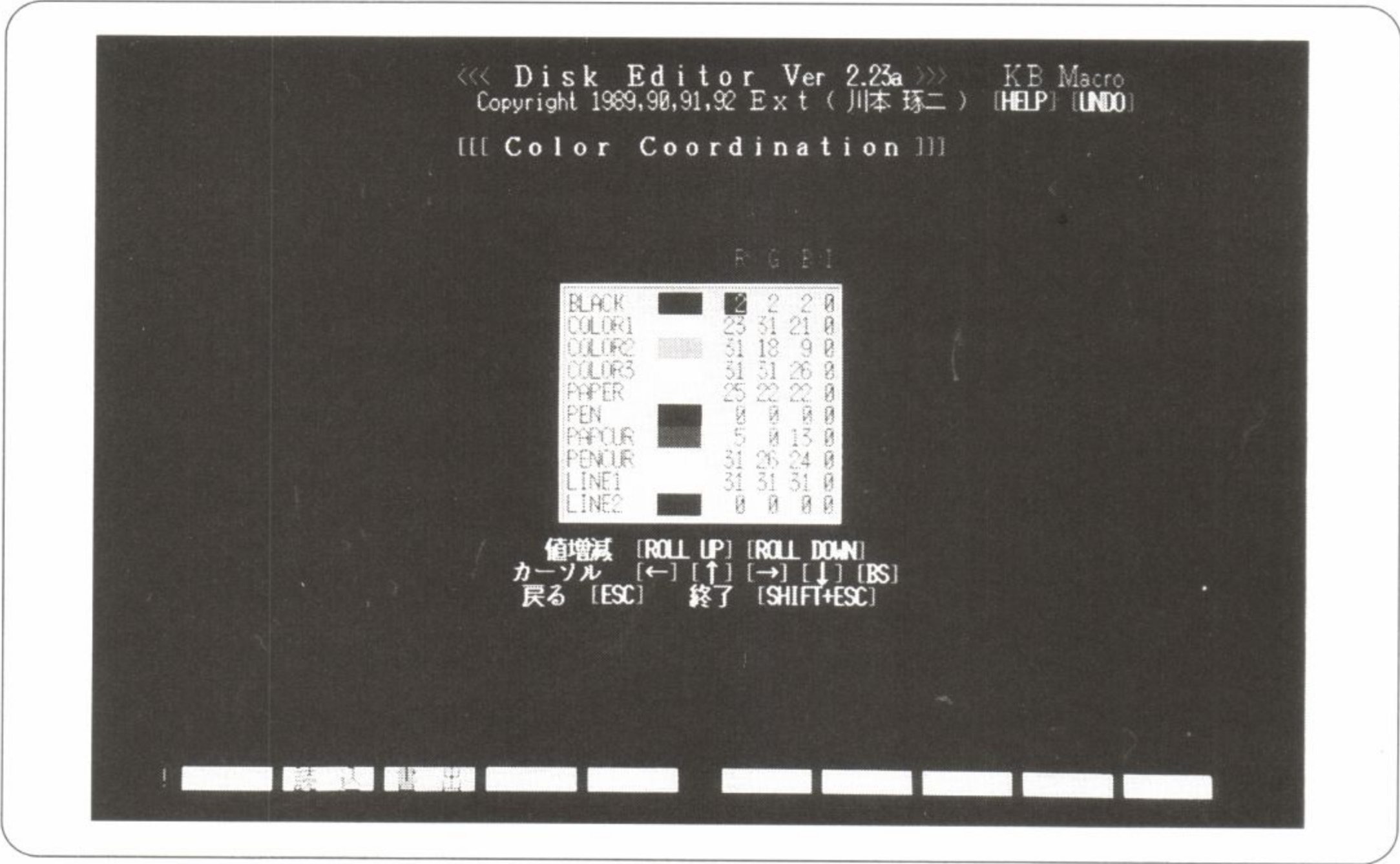
なお、メモリエディットはセクタエディットとほぼ同様の画面で、操作も似ていますが、注意しなければならないのは、入力したデータはメモリ上に即座に書き込まれるということです。「書き出し」というフェイルセーフ機能がないので、システム領域やRAMディスクの領域では、不用意に内容を変更しないよう十分注意して操作する必要があります。

#### ●メモリエディットの使用例

メモリエディットで、X68000のテキストVRAMにデータを書き込んでみましょう。テキストVRAMは、16進でE00000番地から始まっています。[F8](アドレス)キーを押して、アドレスE00000を入力します。あとはデータを適当に入力していくと、画面の左上から入力したデータがビットパターンで表示されます。X68000は、テキストVRAMの１ビットが画面の１ドットに対応するビットマップ方式なので、このようになります。

### 5．カラーを変更する場合

カラーの変更は、deditの画面の配色を変更するものです。メインメニューから「6 coordination」を選択するか、“dedit -c”で起動します。Pht.8がカラー変更画面です。

Pht.8　カラー変更画面

色を変更できるのは、以下の部分です。それぞれRGBの三原色および輝度Iの4つのパラメータを持っています。

| | |
|---|---|
| BLACK | 画面の背景色 |
| COLOR1 | タイトル文字色 |
| COLOR2 | その他の文字色 |
| COLOR3 | 文字色 |
| PAPER | セクタエディットやFATダンプ画面などのウィンドウの背景色 |
| PEN | ウィンドウ内の文字色 |
| PAPCUR | カーソルが点滅するときのカーソル上の文字色 |
| PENCUR | カーソルが点滅するときのカーソル自身の色 |
| LINE1 | ウィンドウの枠の色 |
| LINE2 | ウィンドウの影の色 |

変更する場合は、カーソルキーを使って変更したいパラメータのところにカーソルをあわせ、[ROLL UP][ROLL DOWN]キーにより各項目のパラメータを増減させます。パラメータの変更にあわせて画面上の実際の色も変化します。変更した結果は、[F3](書出)キーによりdedit.xを起動したディレクトリにdedit.cooというファイル名で記録されます。deditは、起動時にこのファイルがあれば自動的に読み込むので、以後の使用時でも、変更した色となります。

## 6．マクロ機能を使用する場合

deditのマクロ機能は「キーボードマクロ」と呼ばれるもので、ED.Xのマクロと同様のものです。使用方法は簡単で、マクロ登録モードにしておいて、一連の操作をユーザの手で実行します。あとはマクロの実行により、登録した一連の操作が何度でも実行されます。

### ●マクロの登録

[HELP]キーを押すと、画面の右上に「KB Macro」の表示が出てキー操作が記録されていきます。再度[HELP]キーを押すと記録が終了します。ただし、記録できるのは100ステップまでのキー操作で、[SHIFT]キーなども1ステップとなります。

### ●マクロの実行

[UNDO]キーにより登録したキー操作が再現されます。

### ●マクロの繰り返し

マクロ登録中に[UNDO]キーを押すと、その時点までに記録されていた操作が繰り返し実行されます。そのままにしておくと、同じ内容が無限に繰り返されますので、中断したいときは[ESC]キーを押します。

**●マクロのセーブ**

マクロ登録中にdeditを終了させようとすると、それまでに登録されたマクロ操作(終了動作も含め)をマクロファイルとしてHuman68kのファイルにセーブするかどうかを聞いてきます。ファイル名を指定することで、マクロファイルを作成できます。

**●マクロファイルの実行**

作成したマクロファイルをdeditの起動と同時に実行する機能です。

dedit -macro=[マクロファイル名]

として実行します。ディレクトリのソートや隙間を詰めるなどの、よく使う機能はマクロファイルにしておくことで、単独のコマンドのように簡単に実行させることができます。

**●マクロの使用例**

**例1）ディレクトリのソート**

dedt223a.Lzhに、dirsort.mcrというファイルが入っていますが、これは、deditのディレクトリエディット機能でディレクトリをソートする操作のマクロファイルです。

dedit -macro=DIRSORT.MCR[CR]

を実行することにより、カレントディレクトリがファイル名の順に並べ替えられます。

**例2）ディレクトリ上の隙間を詰めるマクロを作成する**

ディレクトリエディットで説明したファイル削除でできた隙間を詰める処理です。deditには、このマクロファイルが用意されていませんが、次のようにすれば、簡単にマクロファイルを作ることができます。

| キー操作 | 内　容 |
|---|---|
| dedit[CR] | deditをオプションなしで起動する |
| [HELP] | マクロの登録を開始する |
| [1] | 「1　Human disks」を選択する |
| [3] | 「3　directories」を選択する |
| [SHIFT]+[F10] | 「詰める」を実行する |
| [F3] | 「書出」を実行する |
| [y] | ディスクに書き出してよいか聞いてくるので、yと答える |
| [SHIFT]+[ESC] | deditを終了する |
| [y] | 終了時に、マクロをファイルに記録するか聞いてくるので、yと答える |

| | |
|---|---|
| DIRENT.MCR[CR] | 記録ファイル名を聞いてくるので適当なファイル名（ここではDIRENT.MCRという名前にしているが、ファイル名のつけ方に特に制限はない）を入力する |

これで、ディレクトリ上の隙間を詰める操作がマクロファイルとして記録されました。

**例3）マクロを1つのコマンドのように扱う**

マクロファイルをバッチファイルなどの形で呼び出すようにしたり、HISTORY.Xのalias 機能を利用して、あたかも1つのコマンドのように実行できるようにできます。

まず、適当なディレクトリにマクロファイルを入れておきます。

たとえば、“C:¥bin”に、例1のDIRSORT.MCR、および例2のDIRENT.MCRをコピーしておいたとすると、“dirsort.bat”というファイル名で、

```
dedit -macro=C:¥bin¥DIRSORT.MCR
```

という1行のファイルを作っておけば、dirsort[CR]でカレントディレクトリのディレクトリのソートが行えます。

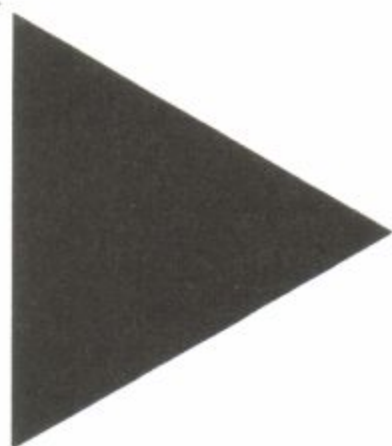
同様に、dirent.batというファイル名で、

```
dedit -macro=C:¥bin¥DIRENT.MCR
```

という1行のファイルを作っておけば、dirsort[CR]でカレントディレクトリのファイル削除でできた隙間を詰めることができます。

HISTORY.Xのalias 機能を使うのであれば、HISTORY.HISファイルのALIASフィールドに、

```
dirsort  dedit -macro=C:¥bin¥DIRSORT.MCR
dirent   dedit -macro=C:¥bin¥DIRENT.MCR
```

を加えておくことで同じように隙間を詰めることができます。

# SRAMCLR.X

## 内蔵SRAMの内容を必要に応じて消去するプログラム

【概要】……………………　X68000の特徴の１つとして内蔵されている16Kバイトのスタティックラム(SRAM)があります。通常、ここにはシステムが起動時に参照するデータが記憶され、起動用のプログラムやRAMディスクとして使われることもあります。これらの内容がなんらかの原因で破壊されたり、ウィルスが潜伏したりすると、システムが正常に起動できなくなります。

そんなときはSRAMの領域を一度すべて消してやればよいのですが、普段使っているときに誤って消されることを防ぐため、ハード的に書き込みが禁止されています。SRAMの領域に書き込むには、特別な手順を踏まないとできないようになっており、少し面倒です。

SRAMCLR.Xは、簡単にSRAMの内容を消去できるだけでなく、必要と思われる領域はそのまま残すといった、選択的な消去も可能です。

【作者】……………………Red FOX

| | |
|---|---|
| PEKIN | REX |
| 梁山泊 | Red　FOX |
| ペンギンランド | PEN16 |
| Cecil-NET | CEL0205 |
| サンデーネット | sun1254 |
| AFH-NETwork | AFH-0561 |
| アスキーネット | PCS28943 |

【作者からの言葉】……　ある日、なにげなくSRAMを覗いたとき、前に消したと思っていたSRAM常駐ソフトがまだ残っているのを見つけました。そこで、これを０で埋めて消してやろうと思って作ったのが、このSRAMCLRです。

で、これをネットにアップしたら、まもなく、ウィルス騒ぎ……。う～ん、なんて、タイムリーなんでしょ。そのウィルスってのが、SRAMに巣食うタイプだったので、ちょうど、このソフトで消せたんですね～。その後、簡単操作を目指してバージョンアップしたものがこれです。

**注意**

SWITCH.Xで何か変更した場合は、SRAMCLRのオプションの６番または９番を実行したあと、変更項目によってはSWITCH.Xの再設定が必要になることがあります。チェックしてみてください。なお、９番を実行する場合は、よく取扱説明書を読んでください。

**【使用する前に】**……… 特に設定すべきことはありませんが、実際にこのプログラムが必要とされるときは、X68000が正常に起動しないといった場合がほとんどでしょう。ご存じのように、X68000は[OPT.1]キーを押しながら起動すれば、メモリスイッチの設定に関係なく、FD→HDの順でシステムを探していきます。ですから、あらかじめ緊急時専用の起動用フロッピーディスクを作成しておき、その中のAUTOEXEC.BATで自動的にSRAMCLRが起動するようにしておくか、パスの通ったディレクトリに入れておくことをお勧めします。

**【主な使用法】**………… メニューを表示して実行する場合は、以下のように実行してください。

SRAMCLR[CR]

**【書式】**…………………… SRAMCLR ［オプションスイッチ……］

**【オプションスイッチ】**

| オプションスイッチ | 内　容 | デフォルト |
|---|---|---|
| -N | テキストパレットをチェックしない | する |
| -P | テキストパレットのチェックのみを行う | |
| -? | オプションスイッチの説明 | |
| -Y | 実行確認をせずに動作する | 実行確認をする |
| -S | SRAM起動になっている場合、4を実行したとき、起動順序を自動的にSTDに変更する | |
| -1 | リザーブエリア以降　[ED0080 - ED3FFF]を消去 | |
| -2 | プログラムエリア以降　[ED0100 - ED3FFF]を消去 | |
| -3 | プログラムエリアのみ　[ED0080 - ED03FF]を消去 | |
| -4 | SRAMDISKエリアのみ　[ED0403 - ED3FFF]を消去<br>(RAM起動時の変更確認あり) | |
| -5 | ROM BOOTアドレスの初期化　[ED000C - ED0010] | |
| -6 | SRAM 全初期化<br>[メモリサイズ] [HD MAX] [SCSI No] [駆動時間] [起動回数]は保存されます | |
| -9 | SRAMすべてを消去 | |

**【使用法】**………………… このプログラムは、SRAMを消去するだけでなく、起動時にSRAMに設定されている画面上の表示色（テキストパレット）をチェックし、異常と判断した場合は標準値をセットします。また、なんらかの原因でSRAM内のテキストパレットのデータが壊れた場合、画面の文字がまったく表示されないことがありますが、このプログラムを実行すれば表示されるようになるので、その後、SWITCH.Xで修正すればよいでしょう。

SRAMCLRをオプションスイッチなしで実行すると、メニューが表示されます(Fig.1)。あとは消去が必要と思われる領域を選んで、その番号を選んでください。

**Fig.1**　SRAMCLRの起動画面

```
SRAMCLR Ver1.8  1992-06-07      By Red ＦＯＸ
          SRAMを消去します(RAM起動になっていた場合、STDに変更します)
                (初期化となっている項目はSRAMに初期値を書き込みます)
                (起動時にSRAMテキストパレット値をチェックします)
                                          [ 実行ｱﾄﾞﾚｽ ]
 1. ﾘｻﾞｰﾌﾞ ｴﾘｱ以降 消去                   [ED0080 - ED3FFF] (お勧め)
 2. ﾌﾟﾛｸﾞﾗﾑ ｴﾘｱ以降 消去                  [ED0100 - ED3FFF]
 3. ﾌﾟﾛｸﾞﾗﾑ ｴﾘｱのみ消去                   [ED0080 - ED03FF]
 4. SRAMDISK ｴﾘｱのみ消去                  [ED0403 - ED3FFF] (RAM起動時の変更確認あり)
 5. 元祖ﾀｲﾌﾟ ROM BOOTｱﾄﾞﾚｽの初期化        [ED000C - ED0010] (起動Error時の表示を戻します)
 6. 元祖ﾀｲﾌﾟ SRAM全初期化   (以下は、保存[ﾒﾓﾘｻｲｽﾞ] [HD MAX] [SCSI No] [駆動時間] [起動回数])
 9. SRAMすべてを消去                      [ED0000 - ED3FFF] (実行後、自動的にリセットします)
                                          (再起動後、switchを設定しなおしてください)
ESC.中止
          どれにしますか？
```

ウィルスが心配な場合などは、1番のリザーブエリア以降の消去を選べばいいでしょう。そのあとで、念のため、SWITCH.Xでブートデバイスの確認をしてください。もしくは6番の全初期化を選びます。この場合、[メモリサイズ][HD MAX][SCSI No][駆動時間][起動回数]は残されますが、その他の設定は初期化されますので、SWITCH.Xでの再設定が必要になります。

9番を選ぶときには十分注意してください。メモリサイズやSASI HDDの台数なども含め、すべてが消去されます。2Mバイト以上のメモリを前提にシステムを構築している場合や、SCSIインタフェースが標準で搭載されていないSUPER以前の機種でSCSI HDDから立ち上げている場合などは、システムが起動できなくなる場合があります。事前に最低限のシステムを入れてあるフロッピーディスクを用意してから実行したほうが安全でしょう。

バッチファイル等でメニューを出さずにSRAMCLRを自動的に実行させたい場合は、起動時にオプションスイッチのどれかをつければメニューは出ません。このとき、無確認で実行させる場合はFig.2のように、“-Y”を先頭につけてください。

Fig.2はリザーブエリア以降の消去を確認を求めずに実行する例です。

**Fig.2**　リザーブエリア以降の消去を確認なしで実行

```
C:>SRAMCLR -Y -1
1. ﾘｻﾞｰﾌﾞ ｴﾘｱ以降 消去                   [ED0080 - ED3FFF] (お勧め)

実行終了しました
```

**COLUMN**

### SRAMのリザーブエリア

X68000が発売された当初と現在とでは、SRAMの使用状況がわずかに変わっています。以前は$ED005B－$ED00FFがシステム予約として使われていませんでしたが、SCSI

ボード、SCSI内蔵機種の登場、SX-WINDOWの発表によって、SRAMのメモリスイッチとしての機能が増えています。出版されている書籍をもとに、追加された機能を簡単にまとめましたので参考にしてください。

| アドレス | 長さ(バイト) | 内　容 |
| --- | --- | --- |
| ED005B | 20 | SYSTEM リザーブ |
| ED006F | 1 | SCSI CHECK DATA |
| ED0070 | 1 | SCSI 本体 ID、内蔵/BOARD フラグ |
| ED0071 | 1 | SCSI SASI ID フラグ |
| ED0072 | 2 | SX-WINDOW CHECK DATA |
| ED0074 | 1 | SX-WINDOW ダブルクリック間隔 |
| ED0075 | 1 | SX-WINDOW マウススピード |
| ED0076 | 1 | SX-WINDOW テキストパレット色相 |
| ED0077 | 1 | SX-WINDOW テキストパレット彩度 |
| ED0078 | 3 | SX-WINDOW テキストパレット明度 |
| ED007B | 1 | SX-WINDOW プリンタドライバ ID |
| ED007C | 1 | SX-WINDOW バージョン、画面保存モード |
| ED007D | 1 | SX-WINDOW DISKTOP PICT ID |
| ED007E | 1 | SX-WINDOW 画面モード |
| ED007F | 129 | SYSTEM リザーブ |

**参考文献**

『Inside X68000』　桒野雅彦 著　ソフトバンク刊
『追補版 SX-WINDOW プログラミング』吉沢正敏 著　ソフトバンク刊

COLUMN

## システムを拡張するプログラム

Human68kは、画面表示を高速化するIOCS.Xや、キーボードからの入力を便利にするHISTORY.Xなどのシャープ純正のプログラムの他、多くのフリーソフトウェアによって、機能や使い勝手を向上させることができます。しかし、ここで紹介するものは、もっと大胆にシステム自体の機能を拡張してしまうプログラムたちです。

**<プログラム名>** TwentyOne
**<作者>** Ext
ファイル名の21文字識別や大文字・小文字の識別、複数のピリオドを使用可能にするなど、Human68kのファイル名の制限をほとんど撤廃します。

**<プログラム名>** lndrv
**<作者>** 沖　勝
UNIXのファイルシステムでは一般的なシンボリックリンクという機能をHuman68kで使えるようにします。シンボリックリンクにより、他のドライブやディレクトリ上のファイル名やディレクトリ名に、別のパス名をつけることができるようになります。
たとえば、“A:¥usr¥x68000¥private¥test”というファイルに対し、“D:¥test”というシンボリックリンクを設定すれば、“D:¥test”というファイルをアクセスすると、自動的に実体である“A:¥usr¥x68000¥private¥test”へのアクセスに変換されます。

**<プログラム名>** execd
**<作者>** 沖　勝
Human68kの実行ファイルには、拡張子として“.x”や“.r”や“.z”がついている必要があります。これは、システムが実行可能かどうかを、拡張子で判断しているからです。execdは、ファイルのアトリビュート領域のうち、未使用のビットを実行可能かどうかの判断として使うことで、拡張子なしの実行ファイルを使用可能にするものです。

**<プログラム名>** scexe
**<作者>** Abechan
スクリプトファイルとは、COMMAND.Xに対するバッチファイルのようなものです。ある実行プログラムに、一連の処理手順を書いたファイル（通常テキストファイルで、これを「スクリプトファイル」と呼ぶ）を読み込みながら処理を行う機能があるとします。このような場合、実行プログラムのファイル名が“abc.x”でスクリプトファイルが“script1”であったとすると、コマンドラインでは、次のようにスクリプトファイル名を実行プログラム名の引き数の形で指定することになります。

```
C:¥>abc script1
```

これに対し、scexeを利用すると、script1の1行目に、そのスクリプトを実行してくれる実行プログラムを、次のような形で書いておくことで、コマンドラインで実行プログラムを明示する必要がなくなります。

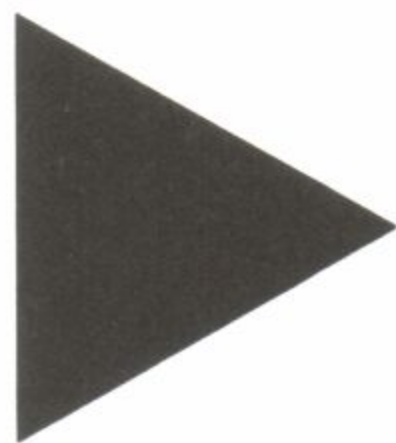
#! C:¥bin¥abc.x
注：scexeは環境変数PATHを見ないので、実行ファイル名はフルパスで指定する。

　これにより、コマンドラインでは、

C:¥＞script1

とすることで、“abc script1”が実行されるようになります。

**<プログラム名> hcommand**
**<作者>** K-ras
　これまで述べたプログラムによる機能拡張は、COMMAND.Xからでは十分に使いこなすことができません。hcommandは、シャープ純正のCOMMAND.Xにパッチを当てて、これらの機能拡張を快適に使用できるようにしたものです。

　これまで述べてきたようなプログラムによる機能拡張は、Human68kの仕様外のため、従来のプログラムではうまく使用できなかったり、不具合が生じたりすることがあります。使用にあたっては、十分注意が必要ですし、なんらかの問題があっても、各自で対処しなければなりません。
　しかし、フリーソフトウェアの作者たちは敏感で、多くの優秀なフリーソフトウェアが、これらの機能拡張プログラムに対応してきています。また、COMMAND.Xに対するhcommand.xのようにシャープ純正のプログラムを対応させてしまうなどの取り組みも行われています。
　本書では、代表例として、TwentyOne.xとhcommand.xを収録しました。他のプログラムについても興味があれば、BBSにアクセスして探してみてください。

# TwentyOne.x

## パワーユーザ御用達のシステム強化プログラム

【概要】……………………　Human68kではファイル名として21文字まで使うことができますが、これはあくまで表示されるだけで、実際に認識されるのは、最初の8文字と拡張子の3文字だけです。したがって、次の2つのファイルは同じ名前と認識されてしまい、同時に使うことができません。

ABCDEFGH1.IJK
ABCDEFGH2.IJK

　たとえば、ABCDEFGH1.IJKというファイルがあるディレクトリに、abcdefgh2.ijkというファイルを新たに作ろうとすると、ファイル名の9文字目の"1"と"2"の違いは無視されて、同じファイル名とみなされて、最初にあったABCDEFGH1.IJKというファイルがabcdefgh2.ijkというファイルに上書きされてしまいます。

　TwentyOneは、この不条理なHuman68kを、その名のとおりファイル名の21文字全部をちゃんと認識するように変更してしまうものです。

　この他、オプションにより、ファイル名の大文字・小文字の識別化や、複数のピリオドを使用可能にするなどの機能があります。「とぅぇにいわん」と呼んでください。

【作者】……………………Ext（川本琢二）　NIFTY-Serve NAH00720
E-Mail　kawamoto@miln.mei.co.jp

【推薦します】…………　X68000は68000というMPUを積んでいるにもかかわらず、そのOSであるHuman68kはMS-DOSと驚くほどよく似ています。そのため、Human68kはMS-DOSを真似たものだといわれるほどですが、メインメモリには搭載したメモリ量だけ最大12Mバイトまで使えます。MS-DOSのように、いくらメモリを搭載しても、OS自体で使用できるメインメモリが制限されてしまうといったことはありません。

　このため、UNIXワークステーションを対象としたGNUなどのソフトウェアが早くからX68000の上に移植されてきました。GCCやTeX、Emacsなどのプログラムが動いたという実績は、まがりなりにもパーソナルワークステーションを標榜したX68000の面目躍如というところでしょう。

注：最近では、MS-DOSマシンでも、80386、486マシンとDOSエクステンダの組み合わせで、これら

のソフトが動くようになっている。

Human68kがファイル名として21文字のうちの８文字＋拡張子３文字までしか識別しないという、明らかにMS-DOSの仕様を引きずっているとしか思えない制限が残っているのは残念なことです。この制限は、UNIXのプログラムを移植する場合の足枷でしかありません。ファイル名にほとんど制限のないUNIXのプログラマたちは、拡張子を含めてもファイル名にたったの11文字しか使えないOSのことなど考えていないからです。

そもそも、UNIXで一般的に使われているアーカイブファイル形式であるtar&compressファイルは、“.tar.Z”という拡張子で、大文字・小文字に複数のピリオドまで含んでいますし、これを展開して出てきたソースファイルも、たった11文字のファイル名ですむことは、まずありません。

こんなとき、TwentyOneがあれば、何の心配もありません。大げさにいえば、TwentyOneは無意味な制限の塊のようなMS-DOSと決別し、パーソナルワークステーションとしてのX68000の本来の姿を見せてくれるソフトなのです。

MAX BBS　BEEPs

**【使用する前に】**………

TwentyOneは、Human68kのシステム自体を変更して機能強化してしまうプログラムです。このため、Human68kのバージョンに依存しており、収録されている TwentyOne Ver 1.23 を使用できるのは、以下のバージョンのHuman68kです。

バージョン 2.02
バージョン 2.03

なお、バージョン 2.03については、HUMAN.SYSのファイルサイズが53626バイトのものと53624バイトのものが存在しています。TwentyOneは、それぞれのバージョンに対応したものを使用しなければなりません。

まず、Human68kの起動時のバージョン表示を確認してください。バージョンが2.02の場合は、TwOn123.lzhにアーカイブされているTwentyOne.xを使用します。バージョンが2.03の場合は、TwentyOne.xに対するバイナリ差分ファイル（本書134ページBdif/Bup参照）が用意されていますので、Bup.xを使用してバージョン2.03用のTwentyOne.xにアップデートしなければなりません。バイナリ差分ファイルは、ファイルサイズが違う２種類のHUMAN.SYSごとに53626.bfdと53624.bfdが用意されていますので、まず、HUMAN.SYSのファイルサイズを確認してください。HUMAN.SYSはシステムファイルのため、DIRコマンドでは表示されませんが、ATTRIBコマンド（コラム「ATTRIBの使用方法」）により、いったんシステム属性を解除してからDIRコマン

ドを使うことで、ファイルサイズを見ることができます。

自分の使っているHUMAN.SYSが53626バイトだった場合は、TwentyOne.xと53626.bfdをカレントディレクトリにコピーしたあと、Bup.xを使用して

Bup 53626.bfd[CR]

とすることにより、ファイルサイズが53626バイトのHUMAN.SYSバージョン2.03用のTwentyOne.xができあがります。

53624バイトだった場合は、53624.bfdを使って

Bup 53624.bfd[CR]

とします。

COLUMN

**ATTRIBの使用方法**

ATTRIBコマンドのシステム属性の操作は隠し機能になっているようで、マニュアルには記述されていませんが、以下のようにすることでシステム属性を解除することができます。

ATTRIB -s HUMAN.SYS

これにより、HUMAN.SYSのシステム属性が解除されます。ファイルサイズを確認したら、以下のようにシステム属性を再設定しておきます。

ATTRIB +s HUMAN.SYS

なお、MF.xやFu.xなどのファイラを使用すれば、そのままシステム属性のファイルも見ることができます。

2.02/2.03以外のバージョンだった場合、残念ながら、TwentyOneを使用することはできません。古いバージョンだった場合は、これを機会にOSをバージョンアップするとよいでしょう。

もし、バージョンがもっと新しいものだった場合は、本書の刊行（1993年3月）以降にHuman68kのバージョンが上がったということです。対応したTwentyOneが開発されているかもしれませんから、パソコン通信に参加して、ぜひ、ご自分で探してみてください。

**【主な使用法】**

TwentyOneはHuman68kのシステムファイルであるHUMAN.SYSに直接パッチを

当てるわけではなく、メモリ上にあるシステムにパッチを当てる方式をとっていますので、使用するときには毎回TwentyOneを実行する必要があります。

とりあえず使用するのであれば、コマンドラインから、

TwentyOne[CR]

とすれば実行できます。これにより、TwentyOneが常駐し、システムにパッチを当ててファイル名を21文字すべて識別するようになります。

この他に、CONFIG.SYSにデバイスドライバとして登録し、システムの起動時に実行させる方法があります。

TwentyOneは、システムの機能強化というプログラムの性質上、使用する場合と、しない場合とに使い分ける意味はないので（コラム「デバイスドライバと常駐コマンド」参照)、デバイスドライバとして最初から登録しておいたほうがいいでしょう。

COLUMN

**デバイスドライバと常駐コマンド**

あとからコマンドラインで起動することが可能なデバイスドライバとしては、OPMドライバ（OPMDRV.X）などがありますが、この場合、音楽関係を使わないときは起動せずにおいて、その分メインメモリを空けておくというメリットがあります。しかし、TwentyOneの場合、システムを機能強化するわけですから、システムの起動時から使えるようになっているべきです。

また、デバイスドライバとして組み込んだ場合はX68000のスーパーバイザ領域と呼ばれるメモリ領域に常駐するので、通常のユーザモードで走るプログラムが暴走しても破壊されることはありません。とはいっても、X68000ではスーパーバイザモードで走るプログラムが少なくありませんから、気休めくらいのものですが。

デバイスドライバとして登録するためには、CONFIG.SYSファイルにエディタ等で次の1行を加えて、再起動します。

device＝TwentyOne.x

もし、TwentyOne.xがCドライブの“¥sys”というディレクトリの下に置いてあるならば、以下のように書きます。

device＝C:¥sys¥TwentyOne.x

**【書式】**……………………TwentyOne［オプションスイッチ］

-B＝<num>　EXBUF の数を<num>にする（補足）　　256

EXBUFというのは、TwentyOneで拡張された機能のためのワークエリアです。このEXBUFが足りなくなった場合、あるはずのファイルが見つからないといったエラーが起こります。通常の使用ではデフォルトの256で足りなくなることはないでしょうが、不都合が起こる場合は、このEXBUFの数を増やしてみるとよいでしょう。

なお、"-B"オプションスイッチの指定は、TwentyOneの常駐開始のとき以外は無効なので、変更した場合は、再起動の必要があります。

| オプションスイッチ | 内　容 | デフォルト |
|---|---|---|
| +T | ファイル名を21文字識別するようにする | +T |
| -T | 8文字＋拡張子3文字の11文字だけの識別にする | |
| +C | ファイル名の大文字と小文字を区別する | |
| -C | ファイル名の大文字と小文字を区別しない | -C |
| +P | ピリオドを拡張子以外にも複数使用できるようにする | |
| -P | ピリオドをファイル名本体と拡張子との区切りにだけ使うようにする | -P |
| +S | "-" や "<" などの特殊キャラクタをファイル名に使用できるようにする | |
| -S | 特殊キャラクタをファイル名に使用できないようにする | -S |
| +D | [CTRL]+[P]キーを押したとき、以後の画面表示をプリンタにも出力するというHuman68kの機能を無効にする | |
| -D | 本来のHuman68kのとおり、プリンタ出力を有効にする | -D |
| +R | "/"から始まるパス名の場合、環境変数"$SYSROOT"の値が先頭に付加される<br>SYSROOT = A:/root となっていた場合に、<br>/etc → A:/root/etc　$SYSROOTの値が付加される<br>¥etc → ¥etc　"¥"の場合は、そのまま扱う<br>B:/etc → B:/etc　ドライブ名があるときもそのまま | |
| -R | 環境変数"$SYSROOT"に関係なく、"/"をルートディレクトリの意味でそのまま扱う | -R |
| +V | オプションの設定状況を表示する | |
| -V | オプションの設定状況を表示しない | -V |

機能によっては、他のプログラムの使用に問題が起こるものもあります。「使用上の注意点」を参照して、各機能の不具合を理解したうえで利用してください。

**COLUMN**……………………

**TwentyOneのオプションの指定**

TwentyOneは、以前のバージョンとオプションの指定方法が一部変わりました。各機能の有効・無効をオプションの＋／－で指定するようになっていますが、以前のバージョンでは、オプションなしで無効、"-"オプションで有効となっていました。

たとえば、複数ピリオドを使うには、以前のバージョンでは"- P"でしたが、現在は、"+P"と指定しなければなりません。以前のバージョンのつもりで"-P"を指定すると、複数ピリオドを使わないという逆の指定になってしまいます。通常の使用では問題ないと思いますが、フリーソフトウェアの中には、TwentyOneの以前のオプションの指定方法を前提としてプログラムされているものがあるかもしれません。もし、そういうプログラムに出会ったら、バグだと早合点しないよう気をつけましょう。そして、もし可能なら、オプションスイッチが変更されたということを作者に報告してあげましょう。

**【使用法】**…………………　コマンドラインもしくはデバイスドライバとして実行することで、TwentyOneがメインメモリに常駐し、オプションで指定した各種の機能を使うことができます。

また、TwentyOneには常駐解除の機能がありませんが、"-B"オプション以外についてはあとから各機能の有効・無効を変更することができ、すべての機能を無効とすることで、事実上TwentyOneが動作していないのと同様の状態に戻すことができます。

そのためには、

TwentyOne -TCPSDR[CR]

とします。

21文字識別を本来の8文字+拡張子3文字だけの識別に戻し、特殊キャラクタについては使用可能とするには、コマンドラインから

TwentyOne -T +S[CR]

とします。

**【使用上の注意点】**……　TwentyOneの全オプションを使うことにより、Human68kでのファイル名の制限はほぼなくなりますが、本来の制限のある仕様を前提とした、従来のプログラムでは、問題が起こる可能性もあります。この場合は、TwentyOne自体の機能をあきらめるか、そのプログラム自体の使用をあきらめ、他のプログラムに乗り換えるしかありません。

ここでは、使用上、問題になりそうな点について紹介します。

**●+T　ファイル名の21文字識別**

この機能が問題になることはあまりありませんが、8文字以上のファイル名をフルに使ったディスクは、他のマシンでは識別できないことになります。もともとHuman68kはファイルフォーマットがMS-DOSと同じため、基本的にはフロッピーディスクに書き込ま

れたデータをファイル変換などをせずに、そのままMS-DOSマシンで使うことができます。

Human68kがファイル名を8文字+拡張子3文字しか識別しないという仕様は、MS-DOSと互換性をとるためと思われますので、ファイル名の21文字識別は、この互換性を捨てることにつながります。その他のオプションによるファイル名の制限の撤廃も同じです。

といっても、もともと完全に互換性があるわけではなく、X68000で小文字のファイル名を使ったり、MS-DOSで“-”を含んだファイル名を使った場合は読み書きできませんから、相互に使うファイルには、これらの文字を使わないといった注意が必要でした。21文字識別を可能にしても、8文字+拡張子3文字で識別できるファイル名を使っている分には問題ありませんので、ちょっと注意することが増えた程度だといえます。

**●+C　ファイル名の大文字・小文字識別**

MS-DOSでは小文字のファイル名をつけようとしても、システムがすべて大文字に直してしまうため、ファイル名に小文字を使うことは不可能です。Human68kでは、このおせっかいな機能がないため、一応ファイル名に小文字を使うことは可能ですが、大文字のファイル名と区別されないため、次の2つのファイルは同じ名前と認識されてしまい、同時に使うことができません。

abcdefgh.ijk
Abcdefgh.ijk

しかし、TwentyOneのオプション“+C”を使うことにより、ファイル名の大文字・小文字を区別することができるようになります。なお、CONやOPMなどのデバイス名に関しては区別しないので、今までどおり大文字でも小文字でも使用できます。

問題となるのは、以下のような場合です。

・コマンド名の大文字・小文字も正確に入力しなければならない。
　例）エディタ“ED.X”を使う場合、今までは、“ED”でも、“ed”でも、“Ed”でも、“eD”でもよかったが、大文字・小文字の認識により、“ED”とタイプしなければならない。

このため、大文字・小文字の識別機能を使う場合は慎重に行う必要があります。

たとえばコマンド名に関してはすべて小文字に統一し、各プログラムが内部で参照するファイル名が大文字か小文字かも調べて、揃えておく必要があります。

しかし、今まではHuman68kが大文字・小文字を区別しない仕様だったために、大文字

・小文字が自由に使われてきましたから、簡単に変更できない場合もあります。この場合は、“＋C”オプションを使うのをあきらめるしかありません。

**●＋P　複数ピリオドの使用**

ピリオドは、ファイル名（8文字）と拡張子（3文字）を区切るための記号でした。このため、

test.tar.Z

というファイル名を指定しても、できるファイルは、

test.tar

というファイル名になってしまいました。しかし、“＋P”オプションでは、通常のファイル名として複数のピリオドが使用できるようになり、このような不具合は解消されます。

問題点としては、COMMAND.Xなど多くのプログラムが、ファイル名の中のピリオドを1つと仮定していることです。このため、DIRコマンドで複数ピリオドのファイルを表示した場合、次のようにおかしな表示になってしまいます。

```
test.tar.Z    →  test      tar
.emacs        →            ema
```

本書には、COMMAND.Xのこのような不都合に対し、COMMAND.Xにパッチを当てて、これらの不都合を解消してしまうhcommnad.x（K-ras氏）が収録されています。これを使えば、通常は複数ピリオドの使用にはほとんど問題がなくなります。

**●＋S　特殊キャラクタの使用**

MS-DOSではファイル名に“-”の文字を使うことが可能なため、MS-DOSマシンで“-”をファイル名に使ったファイルを作った場合、X68000では読めませんでした。しかし、TwentyOneのオプションスイッチ“＋S”により、“-”も含め、ほとんどの文字が使えるようになります。

注：Human68k Ver.2.03からファイル名の途中なら“-”が使えるようになった。

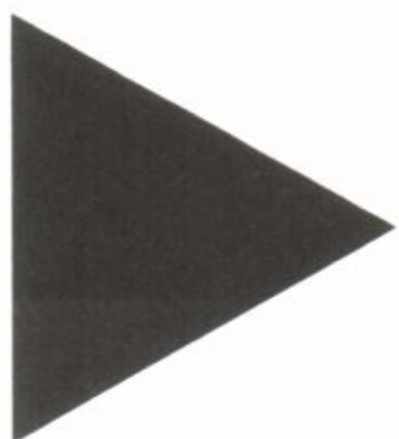
**●＋D　[CTRL]＋[P]キーでのプリンタ出力の禁止**

[CTRL]＋[P]キーによって、以後の画面出力をプリンタにも出力するようにする機能

は、8ビットCPUのOSであるCP/Mの時代から引き継がれているものですが、今や、この機能を必要とする状況はほとんどないといっていいでしょう。むしろ、不注意で[CTRL]+[P]キーを押してしまったとき、プリンタが勝手に動いてしまうほうが不都合といえます。エディタなどのプログラムによっては、この機能が働かないようにするために、わざわざ苦労している場合もあります。

“+D”は、このおせっかいなOSの機能を止めるためにありますが、もしプリンタ出力が本当に必要なら、そのときだけ“-D”で、この機能を有効にすればいいでしょう。

### ●+R　環境変数“$SYSROOT”の使用

TwentyOneは、常駐開始時およびスイッチ変更などでTwentyOne.xを起動したときに読み込みます。したがって、$SYSROOTの値を変更した場合は、再度TwentyOneを起動して新しい$SYSROOTの値を読み込ませる必要があります。

# hcommand.x

## TwentyOneなどのシステム拡張機能に対応させたCOMMAND.X

**【概要】**……………………　X68000のオペレーティングシステムであるHuman68kは、もともとMS-DOSと大変よく似たものでしたが、バージョンアップを重ね、マルチタスク機能や、HISTORY.X、IOCS.Xなどの便利なデバイスドライバの追加により使いやすいものになってきました。

基本的には、フリーソフトウェアは、これらのシャープの環境の上に、さらに使いやすいものを目指して作られていますが、TwentyOneやlndrvなどのように、シャープの提供する環境自体に手を加えてしまったものがあります。これらは、革新的に機能を向上させますが、純正の環境を超えた存在であり、もはやシャープ純正のCOMMAND.Xでは、この機能を使いこなせません。hcommand.xは、COMMAND.Xにパッチを当てて、この拡張された環境に追従するようにしたものです。「えっちなCOMMAND.X」と、呼んでください。

**【作者】**……………………K-ras

| | |
|---|---|
| NIFTY-Serve | HFG02505 |
| 梁山泊 | K-ras |
| KAZUKUN-NET | kazu0011 |

**【作者からの言葉】**……　とあるオフ会での話から作る(?)ことになったhcommand.xですが、自分でも使いづらかったパス名に"/"が使用できないことや、TwentyOne、lndrv、execdがうまく使えないなどということはなんとかクリアしています。結構、おたっきーな改造になってますが、fish、bash、kshなど、もっと優れたshellはたくさんありますので、ぜひ使ってみるとよいのではないでしょうか。

**【推薦します】**…………　TwentyOneは、ファイル名の制限がほとんどなくなるので非常に便利ですが、マルチピリオドにすると、COMMAND.Xがまともに使えなくなってしまいます。UNIXコマンドを使っていましたが、COMMAND.Xを使い慣れている身にはいまいちシックリきませんし、全部外部プログラムになるので、起動が遅くなってしまいます。そんな不便さがhcommand.x によって一気に解消されました。COMMAND.X愛好者には必須でしょう。

MAX BBS　BEEPs

**【使用する前に】**……… hcommand.xは、COMMAND.Xに対するバイナリパッチファイルhcommand.bfdとして配布されています。このため、最初にCOMMAND.Xにパッチを当ててhcommand.xを作る必要があります。

なお、対応するCOMMAND.Xは、

| ファイル名 | 日付 | 時間 | ファイルサイズ |
|---|---|---|---|
| COMMAND.X | 90/05/05 | 12：00：00 | 28026 |

のものだけです。このCOMMAND.Xと添付ディスクのhcmd_05.Lzh（hcommand.bfdを展開しておく必要はありません）がカレントディレクトリにある状態で、

bup　hcmd_05.Lzh[CR]

とします。なお、バイナリパッチを当てるバイナリ差分アップデートプログラムBupについては、本書の134ページを参照してください。これにより、カレントディレクトリにhcommand.xが作成されます。

**【主な使用法】**…………

### 1．コマンドラインからの実行

コマンドラインから実行する場合、

hcommand[CR]

とします。以後は、hcommand.xがコマンドプロセッサとなります。

### 2．起動シェルとして実行

1.の場合は、起動時にCOMMAND.Xが動作し、途中からhcommand.xが動作することになりますが、CONFIG.SYSに次のようにSHELL行を追加することで、最初からhcommand.xを起動シェルとして使用することもできます。

shell＝hcommand.x -P

“P”は、起動シェルとする場合のオプションスイッチで、これによりAUTOEXEC.BATが起動時に実行され、EXITコマンドを実行してもhcommand.xが終了しないようになります。

### 3．COMMAND.Xに置き換えて実行

hcommand.xは、COMMAND.Xのアッパーコンパチです。このため、COMMAND.Xに置き換えてしまうことが可能です。たとえば、純正のCOMMAND.Xをcommand_sharp.xにリネームしておき（いっそ削除してもよい）、hcommand.xをCOMMAND.Xというファイル名にリネームします。これにより、基本的にCOMMAND.Xが起動されるべきところでは、すべてhcommand.xが動作することになります。ED.Xの中から[ESC]＋[C]キーでHuman68kのコマンドを実行する場合などのように、あるアプリケーションの中からコマンドプロセッサを起動する場合、通常はCOMMAND.Xというファイル名で起動されます。このようなときでも、純正のCOMMAND.Xではなく、hcommand.xの機能が利用できるようにするためには、hcommand.xのファイル名をCOMMAND.Xにしておかなければなりません。

なお、hcommand.xをCOMMAND.Xにリネームしておいた場合でも、起動シェルの指定としてCONFIG.SYSにSHELL行を指定してください。これは、SHELL行がない場合、Human68kが起動シェルとしてCOMMAND.Xを“/P”スイッチで起動するのですが、hcommand.xは“-P”でないとオプションスイッチとして受け付けないからです。

```
shell=COMMAND.X -P
```

**【書式】**…………………… hcommand［オプションスイッチ］

**【オプションスイッチ】** COMMAND.Xはオプションスイッチに“/”をつけて指定していましたが、hcommand.xでは“/”をパスの区切りとして使用できるようになったため、スイッチの指定は基本的に“-”で行うようになっています。したがって、以下のCOMMAND.Xの“/スイッチ”は、hcommand.xでは、“-スイッチ”で指定するようにします。ただし、指定文字列を実行するスイッチ“/C”だけは認識しますので、既存のプログラムでCOMMAND.Xを子プロセスで実行する場合でも互換性があります。

| COMMAND.Xのオプションスイッチ | hcommand.xのオプションスイッチ | 内　容 |
|---|---|---|
| /P | -P | AUTOEXEC.BATの実行とEXITを禁止する |
| /D | -D | “-P”とともに指定することでAUTOEXEC.BATの実行をしないようにする |
| /E:＜環境エリアサイズ＞ | -E:＜環境エリアサイズ＞ | 指定した環境エリアサイズを確保する |
| /H:＜ヒストリエリアサイズ＞ | -H:＜ヒストリエリアサイズ＞ | 指定したヒストリエリアサイズを確保する |
| /C ＜文字列＞ | -C ＜文字列＞または /C ＜文字列＞ | 文字列で指定されたコマンドを実行する |

| | | |
|---|---|---|
| | -0 | ログインシェルとして使用する（詳細は「拡張点1．ログイン、ログアウトスクリプトを使用できる」を参照）注 |

注：このスイッチは、hcommand.x独自の機能で、COMMAND.Xには対応するものはない。

【拡張点】　hcommand.xは、通常のCOMMAND.Xとほぼ同様に使用できますが、機能拡張されている点がいくつかあります。ここでは、これらの変更点について説明します。

### 1．ログイン、ログアウトスクリプトを使用できる

hcommand.xを“-0”スイッチを指定して起動した場合、特定のファイル名のバッチファイルを実行します。このファイルは「ログインスクリプト」と呼ばれ、次のファイル名を順に探して、最初に見つかったものが実行されます。

```
.hcmdrc.bat
_hcmdrc.bat
hcmdrc.bat
AUTOEXEC.BAT
```

探すディレクトリは、環境変数“HOME”の指定があればそのディレクトリから、なければカレントディレクトリから、となります。

“-P”スイッチで起動した場合は、カレントディレクトリのAUTOEXEC.BATが実行されますが、hcommand.xはこれをより汎用的にしたものといえるでしょう。

“-P”と違い、“-0”ではEXITを実行することでhcommand.xを終了することができ、「ログアウトスクリプト」と呼ばれるバッチファイルを実行します。このファイル名は、環境変数“HOME”のディレクトリから次の順に探します。

```
.hcmdout.bat
_hcmdout.bat
hcmdout.bat
```

ログアウトスクリプトには、RAMディスク上のデータを保存するとか、ディスクのバックアップを行うなどの定型処理を書いておき、作業の区切りや終了時にログアウトしてから電源オフするといった使い方が有効です。

### 2. パス名の区切りに“¥”に加え、“/”も使うことができる

```
DIR　A:/sys
CHDIR　/tools/cmd
```

といった指定が可能です。

CHDIR、MKDIR、RMDIR、DIR、DEL、REN、COPYといったパス名を指定する部分だけでなく、PATHでのパス名指定や、PROMPTコマンドでの“$p”によるパス名の指定でも“/”が使用できます。

### 3. スイッチの指定が“/スイッチ”から“-スイッチ”に変更 COPYコマンドの連結指定“+”が使えない

COMMAND.Xでは内部コマンドのスイッチとして“/スイッチ”を使用していましたが、hcommand.xでは“/”をパス名の区切りとして使用できるようになった関係で、“-スイッチ”と指定しなければならなくなりました。

たとえば、ディレクトリの表示で横に4つずつ表示させる場合、

```
DIR /W
```

は、“/W”というパス名とみなされてしまいます。

```
DIR -W
```

と“-スイッチ”の形で指定しなければなりません。

その他の内部コマンドのスイッチも同様です。

| COMMAND.Xのスイッチ | hcommand.xのスイッチ |
|---|---|
| COPY [/V][/Q] | COPY [-V][-Q] |
| DEL [/Y][/Q] | DEL [-Y][-Q] |
| DIR [/N][/L][/T][/R][/P][/W] | DIR [-N][-L][-T][-R][-P][-W] |
| REN [/Q] | REN [-Q] |
| VOL [/S][/C] | VOL [-S][-C] |
| HIS [/B] | HIS [-B] |

すべて“-スイッチ”の形式で指定するようになっています。

なお、ファイル名のあとにスイッチをつける場合、

```
DIR ＊.lzh-W
```

とすると、“＊.lzh-w”というファイル名の指定となってしまうので、正しく

```
DIR ＊.lzh -W
```

というように、“-スイッチ”の前をスペースで区切らなければなりません。

もう1つの制限として、COMMAND.XのCOPYコマンドには複数ファイルを“＋”でつなげて書くと連結してコピーするという機能がありますが、hcommand.xでは、“＋”をファイル名として認識してしまうため，連結の指定ができなくなっています。

### 4．TwentyOneのマルチピリオドに対応しているDIRに“-a”スイッチが追加された

Human68kのもともとの仕様では、ファイル名をファイル名本体＋拡張子とし、その区切りとしてのみピリオドが使用されていました。TwentyOneのマルチピリオドでは、ファイル名に自由にピリオドを使用できるようになりますが、純正COMMAND.Xでは、これをうまく扱うことができず、DIR表示がおかしかったり、「ファイルがありません」としてはじかれてしまいます。hcommand.xでは、これらの不具合が修正されています。

さらに、ファイル名の先頭がピリオドで始まっている場合、これを不可視ファイルとみなしてディレクトリ表示しないようになっています。これは、UNIXで行われている慣習で、UNIXから移植されたソフトが多いX68000もこれに従っているわけです。これらの不可視ファイルも含めて表示させたい場合のために、DIRに“-a”スイッチが追加されています。

例）emacs.tar.Z、emacs.x、.emacs、.emacs_keyというファイルがある場合
　　純正COMMAND.XのDIRで表示すると、

```
RAM_DISK                    G:¥bin
   4 ファイル       1769K Byte 使用中        2316K Byte 使用可能
 ファイル使用量     1486K Byte 使用
emacs        tar        501234   92-12-10   20:15:30
emacs        x          837592   92-12-20   22:40:40
ema                        226   91-11-03   12:57:18
ema                        712   91-11-03   12:58:20
```

このように、emacs.tar.Zの“.Z”が欠け、.emacs、.emacs_keyも正しく表示されません。また、“type .emacs[CR]”としても「ファイルがありません」とはじかれてしまい

ます。

　これに対し、hcommand.xのDIRで表示した場合

```
RAM_DISK                G:/bin
    2 ファイル    1768K Byte 使用中    2317K Byte 使用可能
 ファイル使用量   1483K Byte 使用
emacs.tar.Z             501234  92-12-10  20:15:30
emacs.x                 837592  92-12-20  22:40:40
```

と、正しく表示されます。先頭がピリオドで始まる.emacs、.emacs_keyは不可視ファイルとみなされるため表示されていませんが、“type .emacs[CR]”と指定すれば、なんら問題なく認識されます。

　不可視のファイルもすべて表示させるには、DIRに追加された“-a”スイッチを使います。この場合、

```
RAM_DISK                G:/bin
    4 ファイル    1768K Byte 使用中    2317K Byte 使用可能
 ファイル使用量   1483K Byte 使用
emacs.tar.Z             501234  92-12-10  20:15:30
emacs.x                 837592  92-12-20  22:40:40
.emacs                     226  91-11-03  12:57:18
.emacs_key                 712  91-11-03  12:58:20
```

　このように、すべて表示されます。

## 5. lndrvに対応した

　シンボリックリンクとは、他のドライブやディレクトリ上にあるファイルを、シンボリックリンクファイルという特別なファイルを通してアクセスできるようにするもので、lndrvは、これを実現する常駐プログラムです。

　たとえば、“C:/emacs/lisp”というディレクトリを“G:/bin/lisp”というシンボリックリンクファイルでリンクした場合、“G:/bin/lisp”へのアクセスはlndrvにより“C:/emacs/lisp”へのアクセスに変換されます。したがって、“G:/bin/lisp”の下のファイルを表示すれば、“C:/emacs/lisp”の下のファイルが表示されますし、“G:/bin/emacs”へCHDIRすれば、実際には“C:/emacs/lisp”にCHDIRすることになります。

このようなファイルアクセスの動作は、lndrvにより実行されますので、COMMAND.Xでも一応シンボリックリンクのファイルの恩恵にあずかることはできますが、hcommand.xでは、DIR表示でシンボリックリンクのファイルについては“<symlink>”と表示されるようになります。

例）G:/bin/lispがC:/emacs/lispにシンボリックリンクを張っている場合
　純正COMMAND.XのDIRで表示した場合、

```
RAM_DISK                G:¥bin
    2 ファイル      1769K Byte 使用中      2316K Byte 使用可能
 ファイル使用量     823K Byte 使用
emacs                   837592  92-12-20  22：40：40
lisp                        13  93-01-17  20：14：24
```

と、lispが通常ファイルとみなされてしまいます。

　hcommand.xのDIRで表示した場合、

```
 RAM_DISK               G:/bin
    2 ファイル      1769K Byte 使用中      2316K Byte 使用可能
 ファイル使用量     819K Byte 使用
emacs                   837592  92-12-20  22：40：40
lisp                 <symlink>  93-01-17  20：14：24
```

と、lispがシンボリックリンクとして認識されます。

## 6．execd対応

フリーソフトウェアのexecd(exec bit driver)は、Human68kのファイルシステムの未使用ビットを実行可能属性とみなし、このビットが立っているファイルは拡張子にかかわらず実行可能ファイルとして実行できるようにするものです。したがって、たとえば“emacs.x”という実行ファイルを“emacs”という名前に変更しても、実行属性が立っていれば実行できるようになります。

ところが、COMMAND.Xは、“emacs[CR]”と入力した場合、“.x”と拡張子を勝手に補って“emacs.x”という実行ファイルを起動しようとするため、execdが使えませんでした。hcommand.xでは、execdに対応しており、“.x”の拡張子のないファイルも実行可能となっています。

### 7．scexe対応
### バッチファイルで“#”で始まる行をコメントとみなすようになった

通常、hcommand.xは“.bat”という拡張子を見てバッチファイルと認識し、自分で読み込んで実行します。これに対し“.x”などの拡張子を持つファイル、もしくはexecdにより拡張される実行属性付きのファイルは実行ファイルとして認識します。

ところが、フリーソフトウェアのscexe(Script Exec driver)を常駐させ、ファイルの行頭で、

```
#! C:/bin/hcommand.x -c
```

というように、新たなhcommand.xをフルパスで指定しておくと、“.bat”の拡張子ではなく、実行ファイルのようなファイル名でバッチファイルを作ることができます。

たとえば、バッチファイルを“test.x”というファイル名にしておけば、hcommand.xはこれをバッチファイルとは思わず、“test.x”という実行ファイルとして起動しようとします。このとき、scexeが働き、行頭の“#!”以降の記述に従い、

```
C:/bin/hcommand.x -c test.x
```

というように新たにhcommand.xを起動するのです。この、hcommand.xは“test.x”をバッチファイルとして実行してくれます。

なお、scexeのこの仕組みに対応するため、hcommnad.xでは、“#”で始まる行をコメントとみなすようになっています。したがって、hcommand.xがtest.xをバッチファイルとして実行する場合にも、行頭に書かれた“#! hcommand.x -c”の部分はコメントとしてバッチの実行においては無視されます。

また、“#”によるコメントは、“echo on”でバッチ実行中でも表示されませんので、“rem”によるコメントと使い分けると有効でしょう。

---

COLUMN……………………

**スクリプト**

ここでは、hcommand.xのバッチファイルについて説明しましたが、ファイルの行頭に他の実行プログラムを指定しておけば、scexeは、それを実行してくれます。

UNIXの世界では、ある実行プログラムに対し、その動作を記述したファイルは「スクリプト」と呼ばれています。これは、バッチファイルを一般化したものといえるでしょう。

スクリプトファイルを扱うことができるものとしては、たとえば、sed(ストリームエディタ、X68000にも移植されている)と呼ばれるプログラムがあります。sedは編集ファイルに対し、スクリプトに指定した処理を施すもので、編集ファイルの文字をすべて小

文字に変換したり、特定の文字だけを置き換えたりといった処理などが簡単に行えます。通常は次のように指定して実行します。

sed -f ＜スクリプトファイル＞ ＜編集ファイル＞

しかし、スクリプトファイルに実行属性を指定し、行頭に、

#! /usr/bin/sed.x -f

と書いておけば、このスクリプトファイルを直接実行できるようになります。
たとえば、スクリプトファイルが“tolower”という名前だったとすれば、

tolower ＜編集ファイル＞

と指定するだけで、

sed -f tolower ＜編集ファイル＞

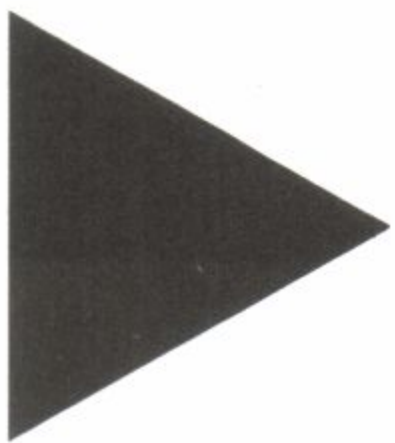
として実行されるのです。あたかも、“tolower”という専用の実行プログラムがあるようなイメージでスクリプトを使うことができます。scexeは、このUNIXのスクリプトの仕組みをX68000で実現したものです。

## 8．sverコマンドが追加された

COMMAND.X自体にもバージョン表示をするVERコマンドがありますが、SVERコマンドはhcommand.xのバージョンを表示します。
例）VERコマンドを実行した場合

Human68k version 2.02

SVERコマンドを実行した場合

hCommand version 2.02(+0.05 Sharp/K-ras)

# CAPS.X

## プログラムごとに自由にキー割り当てを変更、拡張できる常駐プログラム

【概要】

少し前まで、パソコンのソフトウェアの多くはユーザが任意にキーバインド注1を変更する機能を持っていませんでした。ユーザは、そのソフトウェアの製作者が設定したとおりのキーバインドを利用するか、プログラムのソースを書き直して自分の好みのキー設定に変更するしかありませんでした。これでは、ユーザは利用するプログラムごとにキー割り当てを覚えていなければなりません。特にエディタやワープロのように入力と編集を目的としたプログラムでは、使用するソフトウェアが変わるたびに「えっと、このソフトウェアでは、検索はどのキーに割り当てられていたっけ?」などと作業に没頭していた思考をいったん切り替えなければなりませんでした。これでは、作業に集中することはできません。また、キーボードでの入力に慣れた方にはいちいちホームポジションから遠く離れたカーソルキーやファンクションキーに手を伸ばさなければならないソフトウェアというものも嫌われているようです。

X68000の場合、当初から[CAPS]キーがはるかかなたに置かれていたことや、UNIXワークステーションなどの環境に慣れ親しんだユーザが多かったこともあり注2、こういった、キーバインドを変更して、ソフトウェアごとのキー設定の違いを吸収したり、自分の打ちやすい場所にキーを移動させる常駐プログラムが多く公開されていました。

しかし、最近では市販ソフトウェアでもユーザがキー割り当てを変更できるようになり、フリーソフトウェアなどでは、ほぼ同じようなキー割り当てがされるようになってきました。また、フリーソフトウェアの中でもプログラムサイズの大きなものは、ユーザがキー割り当てを自由に変更できるようになっています(キー入力を必要とするようなプログラムがこの機能を持っていないと、ユーザ側からサポートを求める要望が出てくるようです)。こういった事情もあり、単なるキーバインドの変更プログラムは必要性が薄れてきています。

では、CAPS.Xはそれらのキーバインド変更プログラムとどこが違うのでしょうか? ソフトウェアがキーバインドを変更できる環境になっていても、多くのユーザがCAPSを愛用するのはなぜなのでしょう? それは、CAPS.Xが以下のような特徴を持っているからでしょう。

- キーバインドを変更することでキーボードのキー配列を自由に変更できる
- 単純にキーバインドを変更するだけではなく、2ストロークで入力するキーについてもバインドを変更できる注3

- プログラムごとにキーの設定を変更することができる
- FIXER.SYS（シティソフト）と、その機能拡張差分（IKE´氏）[注4]を当てたFIXER.Xにおいては、日本語FEP起動時、文字入力時の変換行におけるカーソルの移動などを独自にキー割り当てすることができる
- 常駐プログラムという形で外部プログラムをメモリ上に配置し、CAPS.X上から値のやりとりや、起動を行うことができる
- 画面の焼き付き防止のために画面の輝度を下げるなどの付属機能を持っている

また、市販ソフトウェアがキーバインドの変更ができるようになったといっても、なにもかも、というわけではありません。たとえば、グラフィックエディタなどではショートカットキーのキーバインドを変更できません。CAPS.Xは、難しい機能や自由なキー割り当てを実現するために、一見して使用頻度の少なそうな機能もたくさんありますが、とりあえずは、プログラムごとのキーバインドの違いを吸収する目的で使用してみてください。それだけでも、CAPS.Xは使って損のないプログラムです。特に「エディタはMicroEmacsに慣れちゃって」という方、「2ストローク入力」をサポートしているのはCAPS.Xくらいです。

**重要！**

CAPS.Xは、基本的にIOCSコールやHUMAN.SYSのワークエリア、コンソール関係のデバイスを参照しています。このため、これらを使用しないプログラム、または乗っ取ってしまうプログラムの中にはCAPS.Xが動作しないものもあります。また、IOCSコールやHUMAN.SYSのワークを参照する都合上、これらのバージョンには注意してください。今のところ、IOCSはバージョンによる問題はありませんが、HUMAN.SYSは「Ver.2.02またはVer.2.03」にのみ対応しています。

また、FIXERの機能拡張差分を当てたプログラム“FIXER.X”では、FIXER.XがFIXER.SYSのキーバインドを変更可能にしています。しかし、同一の機能をCAPS.Xで別のキーに割り当てた場合、FIXER.XのキーバインドよりCAPS.Xのキーバインドのほうが優先されますのでご注意ください。また、常駐の順番はFIXER.Xが先でも、CAPS.Xが先でも正常に動作しますが、常駐プログラムの常識として、常駐解除の順番は常駐させた順番とは逆に行ってください。

注：

1）キーバインド：キーにプログラムなどが持つ機能を割り当てること。

2）UNIXワークステーションに慣れ親しんだ……：UNIXワークステーションでは、ASCII配列のキーボードが多い。JIS配列準拠のX68000キーボードとはアルファベットの配列は同じだが、記号に関しては違った配列になっている。

3）2ストローク入力：2つのキーを順番に押すこと[CTRL]キーを押しながら、他のキーを押すといった操作は1ストロークである。たとえば、エディタでファイルを読み込むとき、ED.Xでは[ESC]、[F]キーの順番に押す。これに対し、フリーソフトのエディタで有名なEmacsやMicroEmacsでは[CTRL]-[X]、[CTRL]-[F]と順に押す。2ストロークキー操作もソフトウェアによってまちまちである。

4）FIXER.SYSのバグとり、および機能拡張の差分ファイルです。ファイル名はFIX120R6.BFD（1993年2月13日現在。作者IKE´氏）で、市販の日本語FEP「FIXER.SYS Ver.1.20」に差分を当てて（差分の当て方については134ページを参照）、FIXER.X Ver.1.20 Rel.6を提供しています。

【作者】…………………… ながい

| | |
|---|---|
| LDB-NET | 1604 |
| NUK-NETWORK | 7699 |
| PECT-NET | 16 |
| FMVAN | 963 |
| スポーツ・南極ネット | 8083 |

【作者からの言葉】…… 私は[CTRL]キーの操作が苦手で、カーソル移動をするのにいちいちホームポジションから離れたカーソルキーを使っていました。CAPS.Xを作るきっかけは、遠すぎる入力モード切替キーを手元で操作したかったからですが、今のように改良した理由は、[CTRL]キーを使わずにカーソルキーをはじめ、すべてのキーをホームポジションに手を置いたまま操作したかったからです。ものぐさな性格がきっかけともいえます。

FIXERに対応したのは、[CTRL]+[XF3]等の入力モード切り替えを無効にしてくれという要望が元になっています。それ以前からFIXER対応と称していたので、対応しているのなら無効にもできるだろうと考えられたようですが、実はFIXERに限らず、CAPS.Xの設定とキーバインドが重なれば、こちらが優先されるに過ぎないものでした。ほとんど詐欺同然です。(^_^;そんなこんなで要望が偏ったこともあって、現在ではすっかりFIXERと切り離せなくなりましたが、あの嘘がなく、要望もなかったら、ここまでやらなかったと思います。妙なもんです、本当。(^_^;

【推薦します】………… 日本の一般的なPCとは配列の異なるSunのWS(ワークステーション)に手が張り付いてしまった私にはもはや手放すことはできません。カーソルキーに手を伸ばすたびに感性をすり減らしているメインキー症候群の人には絶対の自信を持ってお勧めします。特にFIXERやEmacsのユーザには、「黙って使ってみなさい」の一言をあげます。

MAX BBS　An

【使用法】………………… 書式

CAPS.X ［オプションスイッチ］［設定ファイル名］

注：ただし、設定ファイル名が“CAPS.CNF”で、CAPS.Xと同一のディレクトリに設定ファイルが存在する場合は設定ファイル名も省略可能です。

| オプション | 機　能 |
|---|---|
| -M | CAPS.Xが常駐するときに、設定ファイル中でプログラム別のキー設定が設定されている場合、そのプログラム名を表示する。これによって常駐したCAPS.Xが、どのプログラムに対してプログラム別キー設定を働かせるのか、常駐のたびにユーザが確認することができる。“-M”オプションを使えば、このメッセージを画面に出力せずに常駐する。プログラム別にたくさんの設定を書いている方で、いちいち設定内容を確認する必要がない方は、これによって長々と表示されるメッセージをキャンセルすることもできる |
| -R | CAPS.Xの常駐を解除し、キーの設定を元に戻す |

**【主な使用法】**…………　最初に確認しておきますが、CAPS.Xの収録されている圧縮ファイルは“caps061g.Lzh”という名前になっています。また解凍すると、実行ファイルは“CAPS061G.X”となります。これはバージョンの違いをユーザが容易に認識できるばかりでなく、不用意に旧バージョンのファイルが失われることを防止する目的もあります。

圧縮ファイルを解凍すると、その中に“SABUN.LZH”というファイルがあります。これは、CAPS.XのFIXER起動中の機能をFIXER.SYSとFIXER.Xのどちらかに対応させるための差分ファイルです。SABUN.LZHも解凍したうえで、ドキュメントとBdif.x&Bup.xのページをよく読んで、解凍された実行ファイルにどちらかの差分を当ててください。

また、差分を当て終わった実行ファイルは、はじめて使用される方も、旧バージョンを使用されていた方も、“CAPS.X”にリネームしたものとして話を進めます。CAPS.Xに限らず、基本的にバージョンごとに同じプログラムを使い分ける必要もないと思いますので、使用の便宜も考えて同じプログラムは、たとえ別バージョンでも同じファイル名にしたほうがよいでしょう。ただし、きちんと動作することを確認してからにしましょう。なお、言うまでもないと思いますが、旧バージョンもバックアップファイル用のディスクなどに保存しておいたほうがよいでしょう。そのプログラムの新バージョンが“Bdif.x&Bup.x”を使用した旧バージョンへの差分当てという形をとるかもしれないからです。

## 1. とりあえず使用してみる

まず、あなたのディスクの適当なディレクトリに、CAPS.Xとその設定ファイル(圧縮ファイルの解凍直後は、“SAMPLE.CNF”というファイル名のものが付属してきます)をコピーしてください。上記の注意事項の作業は、この作業の前に終了しているものとします。

CAPS.Xは常駐プログラムですし、起動時にAUTOEXEC.BATで登録しないのであれば、できるだけパスの通ったディレクトリに置いておいたほうがよいでしょう。

作業が終了したら、付属のSAMPLE.CNFを使用して、CAPS.Xがどういうプログラムなのか、実際に体験してみましょう。繰り返しますが、CAPS.Xはキーの割り当てを変更したり、キーバインドを拡張する常駐プログラムです。まして、付属のSAMPLE.CNFはキーの割り当てを変更してあるので、あなた自身が使いやすいように変更した「あなたが知っている」キーバインドではなくなります。あるキーを押したら、とんでもないキーを押したことになることも十分考えられます。その点をご了承ください。

まず、CAPS.Xを常駐させます。

CAPS.X SAMPLE.CNF[CR]

(SAMPLE.CNFがCAPS.Xと同一ディレクトリにある場合)

画面にPht.1のようなメッセージが表示され、CAPS.Xが常駐します。

Pht.1　CAPS.Xの常駐メッセージ

```
TwentyOne Ver 1.22a for Human Ver 2.03 Copyright 1991,92 by Ｅｘｔ(T.Kawamoto)

HIOCS version 1.10+14  Copyright 1990,91,92 SHARP / Y.Nakamura

Command version 2.03

A>A:¥BIN¥lndrv.x
A>A:¥BIN¥tfs.x
TFS for X68000 version 1.11a / Copyright 1989-91  卑弥呼☆ MOD By 《雅》
A>
A>PATH=A:¥;A:¥BIN;A:¥SYS
A>path=A:¥;A:¥BIN;A:¥SYS
A>
A>
A>c:
C>cd ¥x68free¥caps
C>caps061g.x caps.cnf
CAPS.x v0.61g 1992 Nagai
Read...caps.cnf
読取中 [!]
読取中 [em.x]
読取中 [mag.r]
読取中 [f.x]
読取中 [sf.x]
読取中 [reorder.x]
読取中 [tf.x]
読取中 [tmn.x]
読取中 [lhv.x]
常駐しました.
C>
```

このメッセージによると、SAMPLE.CNFでは、プログラムごとのキー設定を[em.x](Micro Emacs)、[MAG.r](mag形式のグラフィックローダ)、[f.x][sf.x][tf.x](ファイルユーティリティ類)、[TMN.X](通信プログラム)、[lhv.x](圧縮ファイルを解凍せずにテキストファイルを閲覧するツール)のそれぞれに用意していることがわかります。

CAPS.Xが常駐したら、ちょっと楽しい機能がありますので、見てみましょう。もし、あなたがLEDを内蔵したキー([CAPS]キーなど)を点灯させていたならLEDキーを見てみましょう。点灯させていなかった方は点灯させてみてください。ついでですから、LEDの点灯するキーはすべて点灯させておいたほうがいいでしょう。LEDがゆっくりと点滅していませんか？　まるでキーボードが呼吸しているようです(居眠りかな？)

LEDキーの点灯によってCAPS.Xがきちんと動いていることが確認できたら、キーボードから何か入力してみましょう。通常の英文字はともかく、[CR]キー寄りにあるブラケット([ ])などはうまく入力できませんね。[SHIFT]キーを押しながら、キーボード上部のフルキーの数字キーを押してみてください。[2]を押すと“@”が表示されてしまいました。他のキーも、キートップの表記とは別の文字が画面に表示されてしまいますね。今度は、[かな]キーを押して入力してみましょう。やはり、キートップの表記とは異なる配列ですね。JIS形式のキー配置ではなく、50音順のキーの配列に変更されています。これらは、すべてCAPS.Xが設定ファイルであるSAMPLE.CNFの設定に基づいてキーボードの割り当てを変更したためです。

これでCAPS.Xがどういった働きをするプログラムであるかわかっていただけたかと思います。SAMPLE.CNFの設定は、英数字の入力はASCII配列に、かな入力は50音配列にキーボードの配列を変更しています。

標準のX68000のキー配列に慣れている方には、このSAMPLE.CNFのキー配列変更はとりあえず必要ないでしょう。また、まだまだブラインドタッチができない人にとっても、むやみにキーの配列がキートップに書かれた文字と違ってしまうので混乱するだけです。しかし、普段学校や職場でX68000以外のパソコンを使用し、そのキー配列に慣れてしまった方にとっては、このキー配列の変更機能は大変役立つことでしょう。

それでは付属のSAMPLE.CNFを参考にして、あなたなりのキーバインドの変更を行ってみましょう。SAMPLE.CNFはCAPS.Xが持つたくさんの機能を使いきっているわけではありませんが、主な機能は登録されていますし、設定例として十分参考になります。なによりも整理された設定ファイルなので、これを書き換えていくだけでCAPS.Xをあなたなりの環境で働かせることができるようになります。

実際、ドキュメントを読んでいただくとわかりますが、「とりあえずキー関係でできることはすべてやろう」というような、作者のながい氏の意気込みが反映され、かなりマニアックといってもいいような機能もあります。筆者も、今回の原稿を書く際にはじめて使用した機能もずいぶんありました。日常使用する場合には、これらすべての設定を知っている必要はありませんので、ここでは主要な機能を押さえているSAMPLE.CNFを書き換えていくことを前提に解説していきます。もし他に必要な機能があるかどうかを知りたければ、ながい氏のドキュメントを参照しながら設定してみてください。なかなか読みごたえがある詳しいドキュメントが添付されています。

## 2. デフォルトの設定をする

[!]　　デフォルトの設定

CAPS.Xのキー設定は、大きく分けて「デフォルトの設定」といわれる、基本となるキー設定と、プログラムごとに個別にキー配列を変更する「プログラム別設定」があります。[!]は、ここからデフォルトの設定を始めるという目印のようなもので、この記号以降に{ }でくくられた内容すべてがデフォルトのキー設定となります。プログラム別にキー設定された機能以外は、すべてこのデフォルトの設定が使用されます。

### ●付属機能の設定

SAMPLE.CNFは、まず「付属機能」を設定しています。CAPS.Xのメイン機能は、もちろん、「キーの配列の変更とキー入力の拡張」なのですが、その前にメイン機能をサポートするための設定や、CAPS.Xを使用して得られるおまけ機能の設定を行っています。当然ながら、この機能の設定はデフォルトの設定に含まれています。

| | |
|---|---|
| SHUTTER | 一定の時間（単位：秒）キー入力がないと、画面の輝度を下げてブラウン管の焼き付き注1を防ぐためのもの。設定する場合は数値で設定する。この数値がそのまま入力待ちの秒数となり、設定された時間、キーボードからの入力がない場合は画面を暗くする。この機能をキャンセルしたいときは“OFF”とする<br>例）SHUTTER = 180　（3分間何もキー入力がないと、画面を暗くする） |
| LED | CAPS.X常駐時に、LEDを内蔵したキー（[CAPS]キーなど）のLEDを点滅させるかどうかを、“ON”または“OFF”で設定する<br>例）　LED = ON　（LEDキーを点滅させる） |
| META | メタキーの設定をする。この設定項目で設定されたキーは、単独で入力されると、そのキーのキーコードが出力されるが、設定されたキーを押しながら別のキーを入力すれば、設定されたキーが[ESC]キーのコードを出力し、別に押されたキーが「ESCファンクション」で実現される機能キーとなって出力される。この機能をキャンセルするときは“OFF”とする注2<br>例）　META = XF3　（[XF3]キーにMETAキーを割り当てる） |
| ROMAOFF | この機能に任意のキーを設定することで、設定されたキーが押されている間のみ、ローマ字入力を無効にできる。設定はキーの名前で設定するか、“OFF”で設定をキャンセルするか、で行う<br>例）　ROMAOFF = XF5 |

注：

1）長時間、変化のない画面をディスプレイに表示していると、ブラウン管に焼き付きが起こってしまうことがある。よく店頭などでファンクションキーの部分が画面に残っているディスプレイがあるが、これは画面の焼き付きを起こしたディスプレイである。この機能は、これを防止するためのもの。

2）たとえば、メタキーに[XF3]を設定したとする。これによって、ED.Xを記録して終了させるために、[ESC]キーを押してから[Q]と2回キーを入力していたのが、[XF3]と[Q]を同時に入力するだけで（つまり、1回のキー入力で）操作が終わるようになる。ただし、機能キーとして[SHIFT]キー、[CTRL]キー、[XF1]から[XF5]までのキー、[OPT.1][OPT.2]キーは使用できない。また、入力モードに関係なく、必ず出力は半角英字で出力されるため、たとえば日本語FEPの使用時でも日本語FEPをOFFにすることなく、機能を使用できる。

### ●キー入力拡張機能の全体設定

CAPS.Xのメイン機能の1つで、個別にキー入力拡張の設定をする前に、拡張機能全体の設定を用意するものがあります。

| | |
|---|---|
| FEPCHKEXT | FIXERを起動したときの、通常のキー拡張設定（NORMEXT）の動作を設定する。FIXER起動時に基本設定のキー拡張設定が出力されては困る場合に使用する<br>0 = NORMEXT1とNORMEXT2の両方の基本設定を有効にする<br>1 = NORMEXT1の設定のみ無効にする<br>2 = NORMEXT1とNORMEXT2の両方の設定を無効にする<br>例）　FEPCHKEXT = 0　（FIXER起動時でもNORMEXT1とNORMEXT2の両方の設定を有効にする） |
| KEYBUFCHK | CAPS.Xでキーバッファをチェックし、KEYBUFCHKの設定によってキーボードからの出力のタイミングを調整する（文字落ちなどを防止するため）。値は0〜63を設定できるが、0に設定するとキーバッファが空になるまで次の出力を停止する。63に設定するとキーバッファがいっぱいになるまで、出力し続ける注1<br>例）　KEYBUFCHK = 63 |

| | |
|---|---|
| FIXKIGOMODE | FIXERの記号入力、単語登録、単語削除などの機能を使用しているとき、FIXERが起動したときに使用される設定(FEPWINEXT)を用いるか、変換行での文字列入力時に使用される設定(FIXINPEXT)を用いるかを設定する。FIXER起動時の設定(FEPWINEXT)を使用するときは“WINEXT”を、FIXERで入力作業中の場合の設定(FIXINPEXT)を使用する場合は“INPEXT”に設定する注2<br>例) FIXKIGOMODE = INPEXT (FIXERの記号、単語登録時のキー設定は、FIXERでの変換作業中のキー設定と同じにする) |
| KEYCNGASC | 今までの設定が「キー入力の拡張」機能のための設定だったのに対して、この設定は「キー設定変更」のためにある機能。キー設定を変更してキーコードが変更されたとき、つまり、入力された文字が変更された際、キーコードも変更された文字のキーコードを使用するのか、それともキーコードレベルでは変更までのキーコードとして扱い、出力する文字だけを変更するのかを設定する。アプリケーションなどで直接キーコードを参照しているプログラムでは、キーコードは同じで、最後に出力される文字だけ変更されているのでは正常に動作しないものもある。その場合、キーコードも変更して処理するようになる。通常は“OUT”で使用したほうがよいだろう<br>例) KEYCNGASC = OUT (出力する文字とキーコードは同じにする) |

注:

1) 通常、63に設定しておくだけで問題はない。また、基本的に0と63の間の数値は、ほとんど利用することはないだろう。

2) FIXERを使用するときに、起動時と文字列が変換行に入力されているとき(言い換えれば変換作業中)に、なぜキーの拡張設定を変える必要があるのだろうかと思う人もいるかもしれない。しかし、エディタなどでは、日本語FEP使用時でも、すでに確定した文章を編集したくなるときがあるはず。そういうときに、CAPS.Xは、変換行に文字列がなければ日本語FEPをオン/オフしなくても、CAPSで設定したキー操作が可能になる。また、変換行に文字列があるときは、CAPSで設定したキーで文字列を操作することができる。

### ●キー入力拡張機能の設定

CAPS.Xのメイン機能のうちの1つ、「キー入力の拡張」を設定している部分です。ここで定義された設定が「プログラムごとのキー設定」以外のすべてのプログラムで使用されます。通常のキー設定として「NORMEXT1」と「NORMEXT2」の2つを設定できますが、なぜ、わざわざ標準のキー設定を2つに分けるのかというと、標準的に使用したいキー設定の中でも他のプログラムとキーコードの使用がぶつかってしまい、期待する動作をしない場合があります。そのとき、標準のキー設定をすべてキャンセルするのではなく、一時的にキャンセルしそうなものだけを分けて表記しておくことで、キャンセルされるキー設定をできるだけ少なくしようという目的があります。

### NORMEXT1

**書式:NORMEXT1** {設定}

{設定} ……[in1]+[in2]=[out1]+[out2]

[in?]は入力されるキーコード

[out?]は出力されるキーコード

いわゆる標準でCAPS.Xを使用するときに使用される「キー入力の拡張」を設定します。書式は、{ } でくくって、「入力されたキー」=「CAPS.Xで出力するキー」という形が基本になります。「入力されたキー」と「出力するキー」は、どちらも2つのキーの組み合わせという形をとることができます。

例）NORMEXT1 {ctrl＋shift=caps ctrl＋xf3=ローマ}

例では標準のキー設定として[CTRL]キーと[SHIFT]キーが同時に押された場合、[CAPS]キーをトグルでON/OFFします（ただし、[CTRL]キーが押されてから[SHIFT]キーが押されたことを判定するため、[SHIFT]キーを押してから[CTRL]キーを押しても反応しません)。また、もう1つの設定として、[CTRL]キーが押されてから[XF3]キーが押された場合、[ローマ字]キーがトグルでON/OFFされます。X68000のホームポジションから離れたキーをホームポジション側に移動させる目的で設定されています。

NORMEXT2

**書式**：基本的にNORMEXT1と機能上、設定上の違いはありません。

このように通常の設定を2つに分けずに1つにして設定しておいてもよいのですが、そうすると、何かのキーの設定で、起動したソフトウェア内のキーバインドと重なって不都合があったり、通常は大変便利で重宝しているので設定に加えておきたいのだが、あるソフトウェアでは設定が不要であるというような場合に、通常設定が1つしかないと、他に設定しておきたいものまでその機能をキャンセルしなければなりません。2つに分けておくことで、キャンセルする機能を最小限に押さえることができます。つまり、「NORMEXT1の設定はそのままだが、NORMEXT2の機能はOFFする」などの指定ができるということです。

FIXINPEXT

**書式**：FIXINPEXT {設定}

{設定}はNORMEXTと同様の書式

FIXERを起動し、変換行に文字列が入力されている場合、使用されるキー設定を設定します。つまり、変換中の文字列へのキー操作を設定する機能です。

例）FIXINPEXT {ctrl＋/=xf3}

例では[CTRL]キーが押されて[/]キーが押されたとき、[XF3]（通常のFIXERでは漢

字変換の機能が割り当てられている）を出力します。

FIXWINEXT
書式：FIXWINEXT {設定}
{設定}はNORMEXTと同様の書式

FIXERが起動されたときに使用されるキー設定です。FIXER起動中は、通常はFIXERがキー入力を乗っ取っていますが、この設定をすることでFIXER起動中でも独自のキー設定を使用することができます。
例）FIXWINEXT {ctrl+bs=shift+esc}

例では[CTRL]キーが押されて[BS]キーが押された場合、[SHIFT]キーと[ESC]キーが押されたように出力します（通常のFIXERでは、[SHIFT]+[ESC]キーには「再変換」の機能が割り当てられています）。

**●キー設定変更機能の設定**

NORMKEYCNG
書式：NORMKEYCNG {設定}
{設定}はNORMEXTと同様の書式

標準で使用されるキーコードの変更設定です。NORMEXTとは異なり、あるキーコードを別のキーコードに変更するだけの機能です。
例）NORMKEYCNG {xf1=shift h=i}

この例では[XF1]キーに[SHIFT]キーと同様の機能を与えています。その次の設定は特に意味はないのですが、[H]キーを押すと[I]キーになってしまいます。

FEPKEYCNG
書式：FEPKEYCNG {設定}
{設定}はNORMEXTと同様の書式

FIXER起動時のキーコードを変更します。
例）FEPKEYCNG {opt1=shift}

[OPT.1]キーに[SHIFT]キーのキーコードを割り当てます。

**●アスキーコードの設定**

キーボードから出力されるキーコードを、別に設定されたキーコードに変換します。サンプルの設定を参考にして、自分の好みの配列に書き換えてください。もし通常のX68000のキー配列を使用したい方は(そういう方のほうが多そうですが)、この部分をすべて削除するか、または[#]キーを行の先頭につけてコメント化してください。

設定は、上からそれぞれ「EIJI」(半角英字)、「EIJIS」([SHIFT]キー押下状態での半角英字)、「ROMA」(ローマ字入力)、「ROMAS」([SHIFT]キー押下状態でのローマ字入力)、「KANA」(かな入力)、「KANAS」([SHIFT]キー押下状態でのかな入力)、「CTRL」([CTRL]キー押下状態)という順序で設定されています。それぞれ、そのキーが入力されたときに出力したい文字、またはキーコードを表記してください。

また、文字の前には設定文字であることを示すため、“¥”をつけ、そのまま変更せずに通してしまうキーコードは“¥0”としておいてください。詳しくは“SAMPLE.CNF”をご覧ください。見れば、すぐわかる形になっています。

「ROMA」および「ROMAS」の設定では、文字の前に“¥”ではなく、“!”を設定することで、そのキーのみ、出力を「ローマ字入力」状態ではなく、通常の「半角英字」の文字設定で出力する機能があります。たとえば、ピリオドは「半角英字」では“.”と表記されますが、「ローマ字入力では“。”となります。パソコンでの句読点は「、」「。」を使用せずに「,」「.」で統一している方もいらっしゃるようなので、そういう方は、この設定にしておくとよいでしょう。

## 3. プログラム別のキー設定をする

CAPS.Xが他のキーバインド変更プログラムと異なる点の1つである、プログラムごとのキー設定変更は、起動されるアプリケーション、プログラムごとに異なったキー設定が行える機能です。先述のNORMEXTなどとは異なり、設定されたプログラムの中でのみキー設定が変更されます。

**書式**：[プログラム名] {設定}

{設定}は、今まで説明してきた設定方法と同様です。

例) [tmn.x]

```
        {
          META = OFF
          NORMEXT1{
```

```
                                ctrl+xf5=caps ctrl+.=登録 ctrl+,=shift+登録
ctrl+xf4=ローマ
                         }
                 }
```

プログラムごとにキーバインドが違う場合、すべて統一した設定にしておいたほうが、そのプログラムを使用するときに操作が楽になります。しかし、どうしても他のプログラムとキーバインドが重なってしまう場合などは、それぞれに対応する処理をしておくことで、最小限の変更で同じキーバインドの環境を実現することができます。

筆者もここで1つ例を挙げたいと思います。プログラム別の設定の例として適当ではないかもしれませんが、設定方法の例としてちょっと長めのものを選んでみました。

例）[wp.x]

```
            {
            NORMEXT1 {
                       esc+q=ctrl+clr esc+d=opt1+up esc+a=opt1+down esc+
p=shift+f6
                       esc+t=shift+f8 esc+f=f7 esc+n=f9
                           }
            }
```

これは、X68000本体標準添付のワープロの操作を変更するものです。WP.Xは、Ver.1.10になってコントーロールファンクションが使用できるようになりました。しかし、キーバインドの変更はできません。本体添付のED.Xとコントロールを用いた編集はほぼ同じなので、ED.Xに慣れている方は変更する必要はないかもしれません。

そこで一応、ED.Xとキーの入力を似せるということを目標に設定してみました。コントロールファンクションは同じなので、WP.Xが対応していないエスケープファンクションを再現してみることにします。問題は、エスケープファンクションが2ストローク入力なのに、この設定は同時押下の設定になっていることでしょうか。まったく同じ再現は無理なようですが、CAPS.Xは2ストローク入力もサポートしていますので、工夫次第ではできないこともありません。しかし、同時に押して実現できるのなら、そちらのほうが入力は速いということで、この設定になっています。

また、[ESC]+[N]などは、WP.Xでの機能がプルダウンメニューの中にあり、直接機能を呼び出すことができません。苦しまぎれにメニューを開くだけということにしていますが、CAPS.Xが持つ「キーボードマクロ」機能を使用すれば、「メニューを開いて目的の機能を選択する」といったところまで実現することも不可能ではありません。WP.X側で設定できるファンクションが少なく、ED.Xコンパチとまではいきませんが、プログラム別のキー割り当てがどんなものか、感じはつかんでいただけると思います。

### 4. オプションなどを用いて、ちょっと複雑な設定をする

CAPS.Xでの設定でたびたび述べた{設定}文ですが、SAMPLE.CNFを追っていくうちに“[in1]＋[in2]=[out1]＋[out2]”という、基本の説明にない文字が混じっていることに気づかれたことと思います。たとえば、“‾ctrl＋xf5=caps”とか、“!ctrl＋i=shift＋xf1”などの表記です。CAPS.Xの{設定}文の中では、いくつかのオプションを使用することができます。詳しい説明はドキュメントに載っていますので、ここでは主なものを簡単に表にしてみました。

| | |
|---|---|
| ＊ | out側の先頭に表記。出力される文字をキーバッファには入れず、IOCSのワークのキービットにのみ出力する |
| $0〜$31 | 2ストローク入力を可能にするための一時的なフラグに用いる<br>例）ctrl＋x=$0 $0＋s=esc＋h<br>([CTRL]キーと[X]キーを押し、さらに[S]を入力すると、ED.Xにおいて現在編集中のファイルをセーブし、そのまま編集を続行する） |
| & | in側の先頭に表記。キー入力チェックを、その設定内で使用する同一キーすべてに対し行う<br>例）NORMEXT1＝ {&help=ctrl＋opt.1 help=del}<br>（最初の[HELP]キー入力で出力された[CTRL]＋[OPT.1]が、そのあとに同じキーを使った設定で[DEL]が出力されることで、“[CTRL]＋[OPT.1]＋[DEL]”と出力される） |
| %〜% | キーの連続出力機能。%で囲まれた設定をひと続きの処理として実行する |
| “〜” | 文字列出力。上記の処理と似ている<br>例）opt.1＋h=“Hello!”¥13 |

CAPS.Xは、この他にもさまざまな機能を持っています。基本的な機能は、「キーの設定を拡張する」「キー設定の変更」なのですが、数々のバージョンアップを経て、キー関係の強力な環境プログラムになっています。今まであまり使ったことがないようなプログラムなので、どう使えばいいのかピンとこないかもしれませんが（筆者も、残念ながら、キーバインド以外にはあまりCAPS.Xの機能を使いこなしているとはいえませんが)、うまく設定すれば、今までのキーボード入力に対する考え方が変わりかねないほどのものになりそうです。設定したキーを押せばメッセージが表示され、あるいは通信先へメッセージを送ることなどもできるでしょう。

正直なところ、複雑な設定と、あまり普通のX68000ユーザになじみのないサンプルで損をしているようですが、いろいろいじる楽しみがあり、そして「できることはなんでも取り入れよう」という、作者のながい氏の意気込みが感じられるプログラムです。

また、フリーソフトウェアの中にあっては珍しく、詳しいドキュメントがついてきます。ちょっと内容のつかめない文章もありますが、いろいろ設定を変えてみてください。きっと、新しいキーボードの使い方が見えてくるはずです。

COLUMN

## CAPS.Xのバグ情報

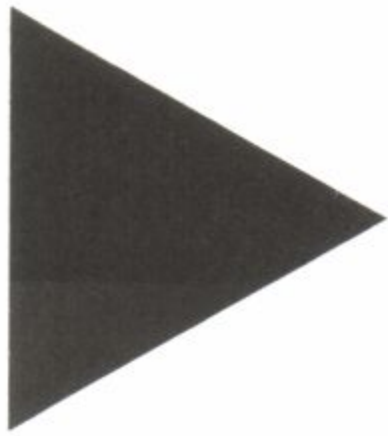
バグ報告がきたものの、作者のX68000が故障して取れなかったバグの症状と回避方法を説明します。

1.ディスプレイシャッター

[A.X] { SHUTTER=180 }
[B.X] { SHUTTER=OFF }

この設定を例にとると、A.XからB.Xの起動後、180秒間設定したキー入力を行わないと、シャッターが動作し、画面が暗くなったまま元に戻らなくなります。めったにないこととは思いますが、この症状が出たときにはジョイスティックかマウスを操作して輝度を戻し、次に適当なキーを押してください。頻繁に起こって困る場合は、とりあえずOFFのかわりに制限時間を長く設定するのもいいでしょう。

2.CTRL {}
まったく機能しませんので、[CTRL]＋[?]の配置変更は拡張設定で行ってください。

なお、以上のバグは、CAPS.Xの、報告されたバグの一部であり、バグのすべてではありません。

# float2p.x

## 純正FLOAT2.X(Ver.2.01)をさらに高速化した浮動小数点演算ドライバ

**【概要】**

MC68000というMPUは、一般に生粋の計算機育ちであるインテル社のCPUに比べ、こと演算においては速度的に劣るとも言われています。X68000においては、プログラムのコーディングの問題、アルゴリズムの問題、基本ソフトの未成熟の問題（デバイスドライバ、OSなどやコンパイラ）などが存在し、速度的にみて「遅い」と言わざるを得ないようです（もちろん、もはや10MHzのMC68000が遅いということは否定できませんが……)。

特に演算ドライバであるFLOAT2.Xは、XCコンパイラやX-BASICでも使われる基本的かつ重要なルーチンでありながら、ルーチンを自作した場合との速度の落差が大きく、実行速度の点で不満のあるものでした。そこで、多くのユーザが純正の演算ドライバに頼らずに実行速度を優先したい場合、自作の演算ルーチンを作ってプログラムに組み込む手間をかけなければなりませんでした。また、そのままFLOAT2.Xを組み込んでいる市販ソフトウェアなどに対しては、速度改善版のFLOATドライバを組み込んでいたのではないかと思います。

これに対し、純正のFLOAT2.XもVer.2になったことで速度面で大きな進歩を遂げました。ここに紹介するfloat2p.xは、この純正FLOAT2.X Ver.2.01をさらに改良することで整数演算を中心に処理速度を上げたものです。

力のあるユーザの中には、「FLOATを呼び出すオーバーヘッド[注]がかかるじゃないか」と言われる人がいるかもしれませんが、自作プログラムでも高速なコーディングをあまりしない、またはそういう暇がない人、とりあえず動くことが大事な人には、逆に「交換するだけで高速化される」現行の方式にも十分な魅力を感じられることと思います。

ともあれ、システムを動かすうえでなくてはならない基本のドライバであり、FLOAT2.Xが組み込まれていないと、X68000の外部コマンド、アプリケーションが動作しないのですから、さらに高速化されたfloat2p.xの登場は、X68000全体の高速化にもつながるはずです。

なお、ネット上では、さらに四則演算を改良して高速化したもの、数値演算プロセッサ用のFLOAT3.Xを改良した、float3p.xも公開されています。

注：あるサブルーチンを呼び出すとき、そのサブルーチンとの間に引数の受け渡しがあります。サブルーチンに処理が移るとき、メインルーチンに戻るときなどです。FLOATの場合、コンパイルされたルーチンは、まず、計算する必要のある数値を手順にそって用意し（レジスタあるいはスタックに数値を用意し)、そのあとに浮動小数点演算ドライバを呼び出します。呼び出された浮動小数点演算ドライバは、まず、ドライバ内の、どのサブルーチンが呼び出されたのかを調べ、そのルーチ

ンに処理を渡します。すると、そのルーチンは受け取った数値を用いて計算を実行し、結果をレジスタに収め、処理をメインルーチンに返します。この一定の手順、演算ごとにわざわざどの処理に振り分けられるのかを調べる動作をMPUがしなければいけないことが、自作で演算ルーチンを組むときの自由度や処理速度（プログラマにはあらかじめ何を計算すればよいのかがわかっているのですから、いちいち演算処理の振り分けを考慮する必要はないわけです。このため、処理の振り分けというプログラムは不要になり、実際に演算処理を実行する速度が向上する結果になります）に比べ、よけいに時間を消費する動作となります。これを、「オーバーヘッド」と呼びます。

【作者】…………………… Pop.（伊野敏之）

| | |
|---|---|
| NIFTY-Serve | GCB02316 |
| サンデーネット | sun2172 |
| PEKIN | POP. |
| 梁山泊 | POP |
| Cecile BBS | CEL0312 |

【作者の言葉】………… このプログラムを作るきっかけですが、実に単純です。その頃、私がアクセスしていたネットの１つで、ちょうど発売されたばかりの「プリンスオブペルシャ」というゲームの話題が出ていました。このゲーム、動きが細かく、なかなかおもしろいのですが、いかんせん速度が遅い代物でした。そこで、ネット上ではすぐさまこれに付属するOPMDRVを改造（音をなくす）して高速化するというプログラムが発表されたのです（ネットでは、よくあることです)。それを見た誰かが､「じゃあ、ペルシャ専用のFLOATを作れば、もっと速くなるのでは？」と書き込みをしました。実際には、そんなことをしてもゲーム中に実数演算をすることはほとんどないので(フライトシミュレータなら別でしょうが)、それは無理な話だったのですが､ちらっとFLOAT関係のIOCSを調べたら、なんと整数演算の命令もあるではないですか(笑)。これは、整数演算部分を高速化したFLOATをアップすればウケるかもしれないと思って作ったのがfloat2pの始まりです。

実際には、たいして速くなかったのですが、予想以上にそのネットでは好評だったので、それならもっと本格的に作ろうと考え、数人の友人とアセンブラレベルでの最適化などを話し合ってできたのが、このプログラムです。正直言って、残念ながら実感できるほどには速くなりませんが、最大の特徴は元のプログラムをほぼそのまま残しているためにバグが少ないという点でしょう。改造版FLOAT2.Xはえてしてバグが多いものなので……。

このプログラムがあなたのX68000で使ってもらえるなら、プログラム作者としてこれほどうれしいことはありません。縁起ものと思ってお使いください(笑)。なお、このプログラムに関しては、著作権を含め、あらゆる権利を放棄していますので、ご自由にご使用ください。同人ソフトなどにも組み込まれているようです。

【使用法】………………… コマンドライン上で

[パス名]float2p.x[CR]

例）実行パスが設定してある場合はそのまま、

float2p.x[CR]

とするか、またはあなたの現在お使いのCONFIG.SYS中の

DEVICE = ¥sys¥FLOAT2.X

を

DEVICE = ¥sys¥float2p.x

のように“P”を付け足し、セーブしたあと、リセットするだけでOKです。

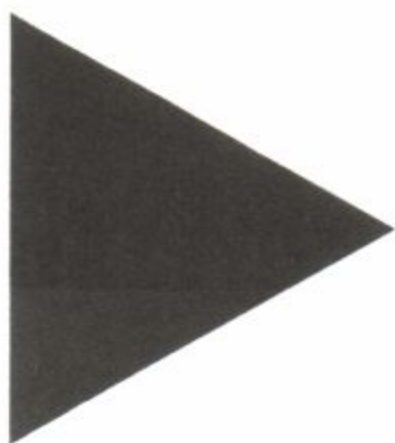
**【ベンチマークテスト】** 最後に、旧来のFLOAT2.X Ver.1.01、FLOAT2.X Ver.2.00、float2p.xのベンチマークテストの結果を載せておきます。使用機種はX68000 PRO2（10MHz）、演算比較はPi.X[注]を使用しています。

注：Pi.X（作者：HIROさん）は円周率を計算し、計算にかかった時間を求めるプログラムです。

```
FLOAT2      X       12844  90-05-05  12：00：00
(EXPERT2およびPRO2同梱のもの。Ver.1.01)

    input a number:100         (小数点以下第100位までの計算)
      3.1415926535897932384626433832795028841971693993751058209749445923
      078164062862089986280348253421170679821480865
    計算時間：182/100 秒

    input a number:200         (小数点以下第200位までの計算)
      3.1415926535897932384626433832795028841971693993751058209749445923
      078164062862089986280348253421170679821480865132823066470938446095
      505822317253594081284811174502841027019385211055596446229489549303
      819644288109
    計算時間：631/100 秒

FLOAT2      X       21956  91-03-15  12：00：00
(XVI同梱のもの。Ver.2.00)
```

```
input a number:100
  3.1415926535897932384626433832795028841971693993751058209749445923
  078164062862089986280348253421170679821480865
計算時間：142/100 秒

input a number:200
  3.1415926535897932384626433832795028841971693993751058209749445923
  0781640628620899862803482534211706798214808651328230664709384460955
  5058223172535940812848111745028410270193852110555964462294895493038
  819644288109
計算時間：490/100 秒

float2p          x          44026  92-09-02    2：00：00
（今回収録のもの。Ver.2.01を改造）

input a number:100
  3.1415926535897932384626433832795028841971693993751058209749445923
  078164062862089986280348253421170679821480865
計算時間：119/100 秒

input a number:200
  3.1415926535897932384626433832795028841971693993751058209749445923
  0781640628620899862803482534211706798214808651328230664709384460955
  5058223172535940812848111745028410270193852110555964462294895493038
  819644288109
計算時間：405/100 秒
```

以上のように、純正のFLOAT2.X Ver.2.01よりfloat2p.xのほうが約2割ほど高速化されています。

| 浮動小数点ドライバ | 小数点以下100位まで | 小数点以下200位まで |
|---|---|---|
| FLOAT2.X Ver.1.01 | 182 | 631 |
| FLOAT2.X Ver.2.00 | 142 | 490 |
| float2p.x | 119 | 405 |

# HIOCS.X

## 高速文字表示が可能なコンソールドライバ

【概要】……………………　HIOCS.Xは、X68000に標準で付属しているIOCS.Xをもとにして、主に画面表示関係の高速化、機能拡張を行ったものです。また、フォントファイルを用意することで全角文字のデザインを変更することができ、通常は使用できない文字コードエリアも使用可能になります。この機能を使えば、NECのPC-9800シリーズ独特の文字（たとえば、丸付数字などのJISで規定されていない記号類）も表示が可能になります。

【作者】……………………YuNK（中村祐一）

| | |
|---|---|
| NIFTY-Serve | GEE01114 |
| TSUKUMO-NET | TSU0630 |
| AFH-NETwork | AFH0343 |
| Milky Way | MILK0018 |
| 梁山泊 | YuNK |

【作者からの言葉】……　決して速いマシンとはいえないX68000ですが、その中で文字表示の速度は普段最も目につくところなので、特にその遅さが目立つと思います。システムディスクには、この文字表示の速度の改善などを目的として、IOCS.Xというドライバが入っています。IOCS.Xは、これはこれで、まあ納得のいく速度なのですが、できるなら、さらにより快適な環境を目指したい、そういう意図で作ったのが、このHIOCS.X です。

ソフトウェアの力のみでできる限りの高速化を目指して各命令のクロック数にまで立ち入っていろいろな工夫をしたのですが、それよりもこのソフトを作るうえで最も苦労したのは、従来のIOCS.Xと置き換えて、いかに不都合が起こらないようにするかという点でした。いろいろ注意はしたのですが、どうしても予想もしなかったようなところに細かい動作の違いが現れてしまうもので、それらの問題点を見つけ出すためには、パソコン通信を通してこのHIOCS.Xを使っていただいているユーザのみなさんのご協力が不可欠でした。そういった意味でも、フリーソフトウェアというのは作者とユーザが協力して作り上げていくものだとあらためて感じますね。

最後に、これからもX68000の機能を生かすようなソフトを作りたいと思いますが、我々ユーザの力でX68000の世界をより活気あるものにしていけたらいいですね。

【推薦します】…………　文字表示が速いぞぉ～！　これが、X68kかぁ？　信じられん。じゃ、比べてみよう。まず、ROM-IOCSでDIRとやってみる。たらたらたら……。IOCS Ver.1.10だと、すすす

っ……。HIOCSでやってみる。パッ！（おぉ）。一度使えば、これなしには、もう68に触れられない!!　HIOCSバンザ～イ！

MAX BBS　Ma-kun

**【使用する前に】**………　デバイスドライバとしてCONFIG.SYSに登録するか、コマンドラインから実行して常駐させてください。フリーソフトのcondrv.sys（X68000の画面表示を高速化する常駐プログラム。表示の高速化の他にバックスクロール機能や、表示フォントの切り替えなど多彩な機能を持っている）と併用するのもよいでしょう。condrv.sysのバックスクロール機能を使用する場合は、condrv.sysのあとに常駐させてください。ただし、この場合、condrv.sysによるフォントの変更機能は無効になります。

注：HIOCS.XのもとになったIOCS.Xとcondrv.sysを併用する場合、IOCS.Xのあとにcondrv.sysを常駐させるが、HIOCS.XをIOCS.Xのかわりに使用する場合は、condrv.sysのあとにHIOCS.Xを常駐させること。

**【使用法】**…………………　デバイスドライバとしてCONFIG.SYSに登録する場合。

DEVICE = HIOCS.X [/オプション……]

コマンドラインから登録する場合。

HIOCS [/オプション……][CR]

HIOCSの組み込みを解除する場合。

HIOCS /R[CR]

**【書式】**……………………HIOCS [オプションスイッチ……]

**【オプションスイッチ】**

| オプションスイッチ | 内　　容 | デフォルト |
|---|---|---|
| /D | DOSコールの画面出力を高速化しない<br>IOCSコールは高速化される | する |
| /G | グラフィックの描画のみ高速化する<br>文字出力は高速化しない | 文字出力も<br>高速化 |
| /MSn | マウスカーソルの移動速度指定(n=0～3)<br>マウスカーソルの移動速度を指定する<br>数字が大きいほど高速になる<br>0を指定すると、マウス関係の処理をROM-IOCSで行う | /MS2 |
| /V | BEEP音をビジュアルベル（画面がフラッシュする）に変更する | 変更しない |

| | | |
|---|---|---|
| /S | 半角文字(¥、‾、|)のスイッチ指定を無視する<br>SWITCH.Xによる半角文字の指定を無視する | 指定に従う |
| /C | 2bytes未定義文字の表示パターンを“※”から任意のキャラクタに変更する（後述） | 変更しない |
| /F[ファイル名] | フォントファイルの指定<br>フォントデータを指定のフォントファイルに切り替える。ファイル名を省略し、“/F”だけの指定ならば ROMフォントを使用、ファイル名を指定するとフォントファイルのデータを使用する。ファイル名を指定するときは、“/F”を省略できる（後述） | ROMフォントを使用 |
| /R | 組み込みを解除する<br>CONFIG.SYSより登録した場合は変更したベクタを戻す<br>コマンドラインから常駐させた場合は、さらに常駐していたメモリの解放も行う | |
| /M | ROM-IOCSに対するパッチ当てのみ行う<br>コマンドラインから常駐させるときだけ有効<br>CONFIG.SYSから登録するときは、このスイッチをつけても他の機能が有効になる<br>パッチ部のみメモリ上に常駐し、常駐解除はできない | |
| /J | 2bytes未定義文字コードのパターン変更をIOCSレベルでサポートしない<br>かな漢字変換にFIXERを使用している場合に指定する | |
| /U | JISコード$2921～$2F7Eの属性を指定する（後述） | |
| /L | JISコード$7621～$7E7Eのフォントモードを指定する(後述) | |
| /E | condrv用エスケープシーケンス[ESC]+[0]、[ESC]+[1]を無効にする(後述) | |

**注**：
1）オプションスイッチの“/”は“-”でもかまわない。
2）オプションスイッチは大文字、小文字どちらも有効である。
3）/M、/J、/E以外のオプションは、組み込み後、コマンドラインからの変更が可能である。
4）また、/V0、/S0のように、最後に“0”をつけることで設定の解除もできる。

**【主な機能】**

## 1. フォントファイルによるフォントの変更

フリーソフトのcondrv.sys、TC.Xと共通のフォントファイルが使用できます。半角文字については、どのソフトも標準のIOCS.Xと共通です。ただし、JISコード$2921～$2F7Eについては扱いが異なりますので、正常に表示されない場合があります。

フォントファイルの構成は、文字コード順にビットパターンをベタに並べたものです。たとえば、“A”という半角文字であれば、

```
00000000 $00
00010000 $10
00101000 $28
01000100 $44
10000010 $82
10000010 $82
10000010 $82
10000010 $82
11111110 $FE
10000010 $82
10000010 $82
10000010 $82
10000010 $82
10000010 $82
00000000 $00
00000000 $00
```

注：$ は16進数を表す。

となりますから、$00、$08、$24……のように1文字分16バイトを文字コード順に並べたものです。もちろん、全角文字ならば、32バイト分になります。

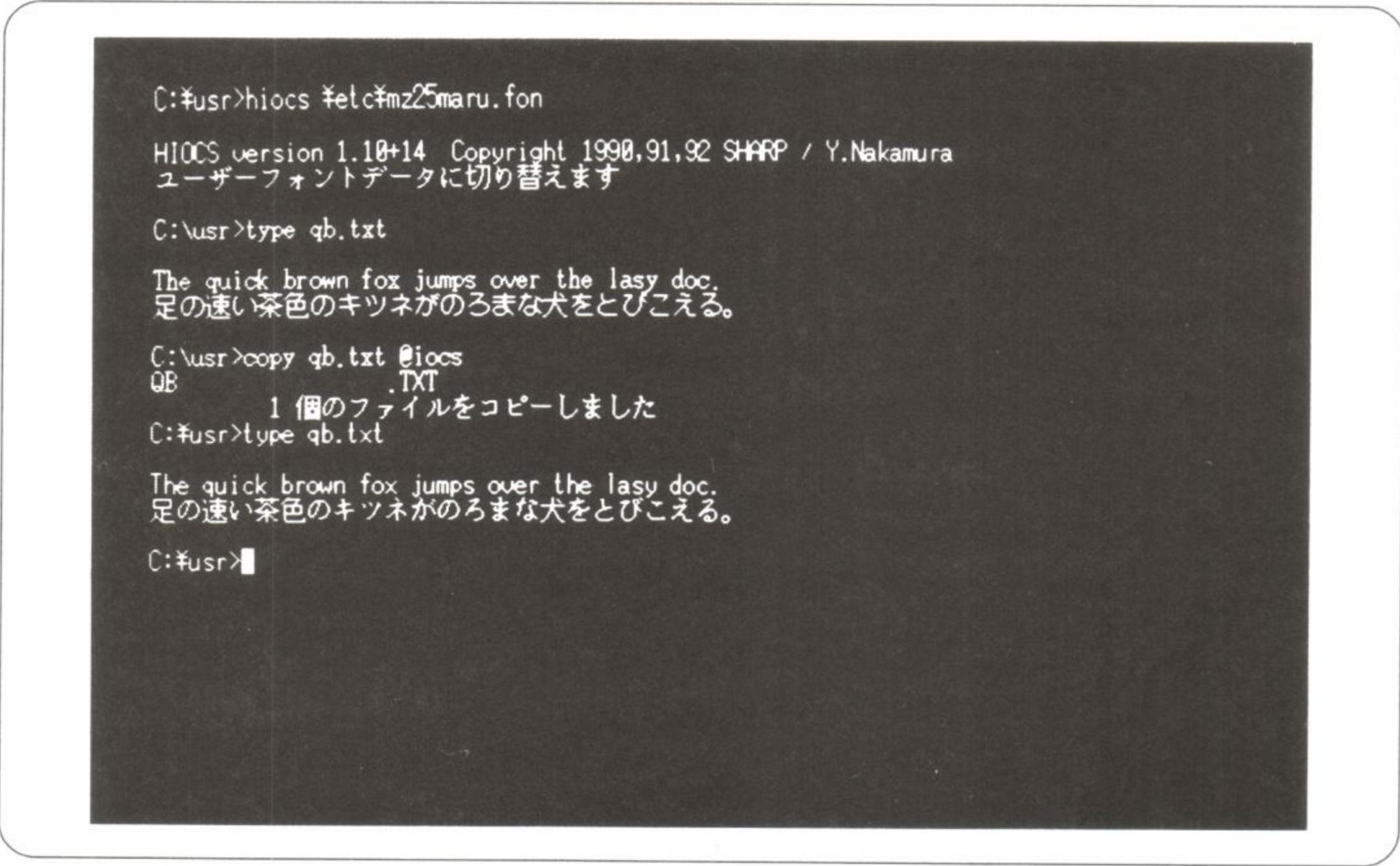
HIOCS.Xでは、以下のサイズのファイルを扱えますが、常駐したときの大きさがその分大きくなります。

| | |
|---|---|
| 半角128文字分(英数文字) | 2048バイト |
| 半角256文字分(英数文字＋半角かな) | 4096バイト |
| 半角256文字＋全角非漢字(JISコード$2921〜$2F7E) | 49216バイト |
| 半角256文字＋全角非漢字・第一水準漢字 | 145472バイト |
| 半角256文字＋全角非漢字・第一、第二水準漢字 | 286848バイト |

また、標準のIOCS.Xでは、CONFIG.SYSからデバイスドライバとして登録したときに限り、@IOCSというデバイス名が使えるようになり、そこにフォントファイルを書き込む(たとえば、C:>COPY FONTFILE @IOCS) ことでフォントの切り替えができるようになっています。

これに対し、HIOCS.Xでは、コマンドラインから常駐させても同様の機能が使えます。また、上述のサイズ以外のファイルを@IOCSに書き込むことで、フォントが初期化され、X68000本体に内蔵されたROMのフォントに戻ります(Pht.1参照)。

Pht.1　フォントファイルによる文字パターンの変更

```
C:¥usr>hiocs ¥etc¥mz25maru.fon

HIOCS version 1.10+14  Copyright 1990,91,92 SHARP / Y.Nakamura
ユーザーフォントデータに切り替えます

C:\usr>type qb.txt

The quick brown fox jumps over the lasy doc.
足の速い茶色のキツネがのろまな犬をとびこえる。

C:\usr>copy qb.txt @iocs
QB              .TXT
        1 個のファイルをコピーしました
C:¥usr>type qb.txt

The quick brown fox jumps over the lasy doc.
足の速い茶色のキツネがのろまな犬をとびこえる。

C:¥usr>
```

## 2．JISコード$2921～$2F7Eの文字属性の指定

JISで規定されている文字コードエリアはすべてが規定されているわけではなく、メーカが自由に規定できる部分がかなりあります。この範囲のコードもそうで、NECのPC-9801では2bytes半角の罫線やJISで決められていない記号類になっており、X68000では表示されません。このため、この部分の文字を表示できるようにするフリーソフトウェアがいくつか発表されています。当然、標準のIOCS.Xとでは文字の扱いが異なってきますので、どちらの仕様にあわせるかを指定します。

| オプションスイッチ | 内　容 |
|---|---|
| /U0 | $2921～$2B7E、$2E21～$2F7Eは未定義（拡張用）となり、$2C21～$2D7Eはシステムの外字領域となる（ROM-IOCS/IOCS.X準拠） |
| /U1 | $2921～$2B7Eはユーザフォントデータを半角で表示し、$2C21～$2F7Eは全角で表示する（TC.X/HST.X/PC-9801準拠） |
| /U2 | $2921～$2F7Eのすべてで、ユーザフォントデータを全角で表示する |

全角非漢字を含むフォントファイルが使用されていない場合、これらの指定は無視されます。オプションスイッチ“/U”のみを指定した場合は、“/U1”と同じ扱いになります。

また、アプリケーションプログラムによっては、“/U”オプションスイッチで文字の属性を変更しても表示できなかったり、通常とは違った動作をすることがありますので注意してください。

## 3．JISコード$7621～$7E7Eのフォントモード

$7621から$7E7Eまではシステム外字のエリアですが、“/L”オプションスイッチを使用することによって、フォントファイル内のデータを使用するかどうかを指定できます。

| オプションスイッチ | 内　容 |
|---|---|
| /L0 | 従来どおり |
| /L1 | $7621～$7E7Eはユーザフォントデータを全角で表示する |

このスイッチを省略した場合は“/L0”とみなされ、“/L”のみの指定ならば“/L1”とみなされます。ただし、この部分はJIS第二水準の漢字コードエリアの最後にあたるため、第二水準漢字までのフォントファイルを使用したときのみ有効になります。それ以外のフォントファイルを使用したときは、このスイッチは無視され、通常のシステム外字が使われます。

## 4．組み込み解除

IOCSコール、DOSコールをユーザが使用するプログラムから使う場合、直接、その動作を行うプログラムの先頭アドレスから実行するのではなく、そのアドレスが書いてあるテーブルをいったん参照してから実行するようになっています（両コールの実際の使用方法については、シャープ(株)から発売されている「Compiler Pro-68k」付属の「プログラマーズマニュアル」等を参照してください）。

このような間接的な呼び出し方をするのは、機能の拡張や修正が、そのテーブルに書いてあるアドレスを変更するだけで簡単に行えるからです。そのアドレスのことを「ベクタ」と呼びます。もちろん、IOCS.X、HIOCS.Xともに、このベクタを自分自身が持っている拡張されたプログラムのアドレスに変更して機能拡張を行います。

"/R"スイッチを使って組み込みを解除する際、CONFIG.SYSからデバイスドライバとして登録した場合と、コマンドラインから常駐させた場合とではメモリの解除のしかたが異なります。CONFIG.SYSからデバイスドライバとして登録した場合は、このベクタを元に戻すだけなので、使用しているメモリはフォントのバッファを含めて使われたままになります。一方、コマンドラインより常駐させた場合はメモリをすべて解放します。

ただし、"/M"スイッチを使用してコマンドラインから常駐し、ROM-IOCSのバグの修正だけを行っている場合は、ベクタも元に戻されませんし、常駐解除もできません。また、CONFIG.SYSから登録したあと、組み込みを解除しても、この修正だけは有効になったままです。

### 5．2bytes未定義文字コードの表示パターン変更

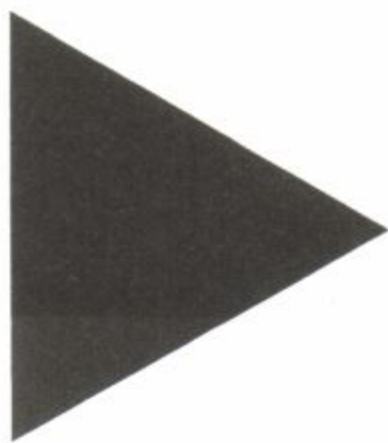
"/C"オプションスイッチのあとに全角文字をつけると、未定義文字の表示パターンが全角文字に変わります。全角文字を省略すると、網状のパターンを表示します。また、任意の文字パターンを設定することが可能です。この場合、"/C"のあとに32バイト分の16進数を続けてください。

```
HOCS /CFFFF0000FFFF0000FFFF0000FFFF0000FFFF0000FFFF0000FFFF0000FFFF0000
```

(2bytes未定義文字の表示パターンが横縞になります)

日本語FEPにFIXERを使用している場合、FIXER内部の処理の都合上、常駐後のベクタ変更ができなくなることがあります。これを避けるため、常駐時に"/J"オプションスイッチを指定してください。

### 6．"/E"オプションスイッチ

フリーソフトウェアのcondrv.sysを使用すると、画面への出力をバッファに保存しておき、あとから参照することが可能になります。また、[ESC]+[0]、[ESC]+[1]を画面に出力することによって、バッファへの取り込みを中断／再開することができます。condrv.sysの"-E"スイッチは、このエスケープシーケンスでのコントロールを無効にする指定です。condrv.sysとHIOCS.Xを併用すると、condrv.sysからはこのオプションが機能しなくなるため、HIOCS.X側で指定します。

# FLEXDISK.SYS

## 高速、再確保可能なRAMディスクドライバ

【概要】

FLEXDISKは、RAMディスクドライバ本体であるFLEXDISK.SYSと、設定後、FLEXDISK.SYSをコントロールするためのプログラムであるFLEXCTRL.Rの2本から構成されます。FLEXDISK.SYSを使うと、コマンドラインからRAMディスクに使っているメモリの解放、再確保が可能です。また、転送速度(コピー等の時間)も純正のRAMDISK.SYSに比べて高速(約5倍)です。

X68000の場合はメモリが非常に高価ですから、グラフィックVRAMをRAMディスクとして使っている人が多いのではないかと思われます。その場合、グラフィックVRAMを使用するようなソフトを使うと、RAMディスクの内容が壊されてしまい、FORMATコマンドでフォーマットし直すしかありませんでした。しかし、このFLEXDISKを使うと、簡単な操作でRAMディスクの再確保が行えます。

RAMディスクというのは縁の下の力持ちのような存在で、使い慣れてしまうと、空気のような感覚で使ってしまうものです。しかし、純正のRAMディスクドライバは、空気のように、というにはちょっとアクセススピードが遅いようです。このFLEXDISKの快適さは、RAMディスクを使用する人には大変ありがたいものです。

ちょっと特殊な例かもしれませんが、筆者の場合、特に重宝しているのは、このFLEXDISKのノーマルモードでのファイルエントリ数（簡単にいえば、ルートディレクトリに作成可能なファイル数のこと）が、FLEXDISKで確保する全体容量が1232Kバイト以上の場合、192ファイルまで作成可能なことです。大きなプログラムを作成するときには、通常、分割コンパイルで行いますが、この場合、プログラムソースを高速に読み書きできるRAMディスクに置くと、コンパイルにかかる時間を短縮することができます。

ところが、標準のRAMディスクドライバを使うとファイルエントリ数が92個までなので、それ以上ファイル数が多くなるとRAMディスクに置くことができなくなります(ただし、これはファイルエントリ数についてなので、ディレクトリを作って、その下に置く場合は92個以上でも可能です)。

そういう点では、このFLEXDISKのファイルエントリ数が192個までなのは大変重宝しています。このような無謀なことは通常の記録媒体においては行わないのですが、筆者の場合、一時的な作業場所として大変使いやすいRAMディスクを、ディレクトリを作らず、このような形で利用しています。

【作者】……………………T.Nishikawa　　MAX BBS　　64 Oh!
サンデーネット　64 devil

【推薦します】…………　君のアクセスランプは眠っていないか！（ぉぃ）　やはり、これが最大の売りでしょう。まったく音がしない便利なRAMディスクも、ときとしては暴走していないか不安になるもの……。そんなときに、このアクセスランプは精神安定剤として大いに役立ちます。なによりもまずかっちょええしね！

MAX BBS　てぃん

【操作方法】……………●FLEXDISK.SYSの設定

FLEXDISKを使う場合は、今使っているRAMディスクドライバのかわりにCONFIG.SYSでFLEXDISK.SYSを指定します。

DEVICE = FLEXDISK.SYS　容量指定

容量の指定は、シャープ純正のRAMDISK.SYSと同様、“#G”や“#M128”といった指定が可能です。通常のRAMディスクとして使用する場合は、これだけで終わりです。標準のRAMDISK.SYSと同じような使い方が可能です。

●FLEXDISK.SYSの特殊な使用方法1

グラフィックVRAMをRAMディスクとして使っている場合に、グラフィックVRAMを使用するようなソフトを使用してしまうと、RAMディスクが壊れてしまいます。その場合でも、FLEXDISKでは、コマンドラインから、

FLEXCTRL /C [CR]

と入力するだけで、新たにRAMディスクを使用することができるようになります。ただし、この場合もRAMディスクが使用可能となるだけで、以前の内容が保証されるわけではありませんので注意が必要です。

●FLEXDISK.SYSの特殊な使用方法2

FLEXDISKが起動する時点で[ESC]キー（もしくは何も押さないか）、[XF1]～[XF5]キーを押すことにより、6種類のFLEXDISKの確保方法が選べます。

CONFIG.SYSで6種類の容量を指定します。

DEVICE = FLEXDISK.SYS [P1][P2][P3][P4][P5][P6]

パラメータの[P1]～[P6]で各キーが押されたときのRAMディスクの容量を指定します。設定した内容を忘れてしまった場合でも、[ESC]、[XF1]～[XF5]のかわりに[HELP]キーを押すことによって、次のようなメニューが表示されます。画面で確認したうえで[ESC]、[XF1]～[XF5]を押しても選択することができます。

```
FLEXIBLE RAMDISK DRIVER X68000 v1.03 Copyright 1991～92 T.Nishikawa
SELECT [ESC]=#M2050 [XF1]=#C [XF2]=#GC [XF3]=#GM720 [XF4]=#G [XF5]=#M720
```

**●FLEXDISK.SYSの特殊使用方法3**

FLEXDISKでは、RAMディスクの確保方法にコンパチブルモードというのがあります。コンパチブルモードを使うと、RAMディスクをHuman68k標準のフォーマットである2HDと同じ仕様で確保することができます。2HDと同じ仕様なので、DISKCOPYコマンドがそのまま使えます。このことは、バックアップ等の一時的なメディアとしてRAMディスクが使えるということを意味します。

**【環境設定】**…………… 環境変数としては、flexdiskがあります。この環境変数は、

FLEXCTRL /E [CR]

のように、オプションスイッチの“/E”をつけて実行することによってFLEXDISKのドライブ名が設定されます。AUTOEXEC.BAT等でテンポラリパスなどを設定する場合に利用できます。

例)

FLEXCTRL /E

TEMP %flexdisk%¥

**【書式】**………………… DEVICE = FLEXDISK.SYS [/オプションスイッチ] [P1] [P2] [P3] [P4] [P5] [P6]

| オプションスイッチ | 内　容 | デフォルト |
|---|---|---|
| /A | RAMディスクにアクセスすると、X68000本体のアクセスランプが点灯するようにする | 点灯しない |
| /C | TIMER-C処理（例外処理注）を高速化する | 高速化しない |

| | | |
|---|---|---|
| /P | キーボード横のマウス端子の検出を省略して、キーボードBIOSを高速化する（X68000-PROタイプのキーボードには、そもそもマウス端子がないのでお勧め）このスイッチについては、“/C”が指定されている場合のみ有効である | 検出する |

注：X68000では、例外処理の一種としてTIMER-Cの処理がある。これを、FLEXDISK.SYSの独自の内部ルーチンで置き換えることにより、高速化を図るものである。ただし、TIMER-Cの例外処理は、多くの常駐タイプのソフトで使われている可能性が高いので、その場合は常駐の順番を変えるか、もしくは“/C”オプションスイッチを使用しないようにしたほうがよいだろう。置き換える内部ルーチンは上位互換となっているので、それほど大きな問題は生じないはずである。

また、[P1]～[P6] については、[初期化モード][確保容量] で指定していきます。

| 初期化モード | 内　容 |
|---|---|
| # | 起動時点において、RAMディスクの状態をチェックし、壊れている場合は自動的に初期化を行う。壊れていない場合は、初期化を行わずに立ち上げる。なお、[SHIFT]キーを押しながら起動すると、強制的に初期化する |
| % | つねに起動時点において強制的に初期化する |

| 確保容量 | 内　容 |
|---|---|
| G | グラフィックVRAMを512Kバイト使用する |
| Mn | メインRAMを nKバイト使用する<br>n＝16K～12000Kバイト |
| C | メインRAMを1232Kバイト使用してコンパチブルモードとする |
| GMn | グラフィックVRAMを512Kバイト、メインRAMを nKバイト使用する |
| GC | グラフィックVRAMを512Kバイト、メインRAMを720Kバイト使用してコンパチブルモードにする |

[P1]～[P6] については、起動時に押されているキーにより設定されます。

| パラメータ | 起動時に押されているキー | デフォルト |
|---|---|---|
| [P1] | 起動時に[ESC]キーもしくは何も押さなかった場合 | #M1232 |
| [P2] | 起動時に[XF1]キーを押していた場合 | #C |
| [P3] | 起動時に[XF2]キーを押していた場合 | #M720 |
| [P4] | 起動時に[XF3]キーを押していた場合 | #GM720 |
| [P5] | 起動時に[XF4]キーを押していた場合 | #GC |
| [P6] | 起動時に[XF5]キーを押していた場合 | #G |

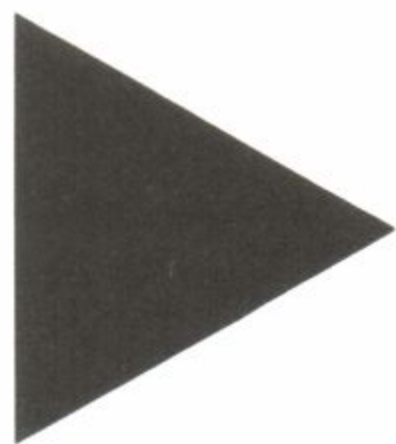
**【使用法】** ………………… FLEXDISK.SYSをコントロールするFLEXCTRL.Rの使い方は、以下のように設定します。

FLEXCTRL [/オプションスイッチ]

| オプションスイッチ | 内　容 |
|---|---|
| /S | ディスクの再確保を行う<br>RAMディスクの状態をチェックし、壊れている場合は初期化を行う |
| /C | ディスクの再確保および初期化を行う<br>RAMディスクがすでに確保されている場合は初期化のみを行う |
| /R | ディスク領域の解放を行う |
| /D | FLEXDISKの情報を表示する |
| /E | FLEXDISKのドライブを環境変数“flexdisk”に設定する |

**【その他】**……………… ちなみに筆者の設定は、次のようになっています。

DEVICE = FLEXDISK.SYS #M1600

これは、起動時点においてRAMディスクの状態をチェックし、壊れている場合は初期化を行う設定です。壊れていない場合は、初期化を行わずに立ち上げます。また、ノーマルモードで通常メモリに1600Kバイトを確保するようになっています。また、TIMER-C処理は高速化せず、タイマーランプの点滅は行わないようになっています。

# DCACHE2.R

## デバイスドライバに割り込んで動作するディスクキャッシュプログラム

【概要】……………………　ディスクキャッシュプログラムとは、CPUやメモリなどに比べ、動作が遅いフロッピーディスクやハードディスクなどの外部記憶装置を少しでも高速に使うためのプログラムです。

あらかじめメモリ上に領域を確保しておき、外部記憶装置とデータをやりとりする場合は、読み込んだデータを溜めておきます。再度同じデータが必要になったときには、外部記憶装置にアクセスせずに、溜めておいたデータを転送します。

このように動作することを「キャッシュ」、または「キャッシング」といい、処理速度の遅い外部記憶装置にアクセスすることを最小限に抑え、効率を上げようとするわけです。

Human68kにもほぼ同様の機能があり、CONFIG.SYSの設定の1つであるBUFFERSでディスクバッファと呼ばれる領域を確保しますが、この領域はもっぱら入出力時に転送するデータのサイズをセクタ（クラスタ）のサイズにあわせたり、FATやディレクトリの読み書きに使われます。

そこでHuman68kが用意するものとは別に、もっと大きな領域を確保してディスクアクセスの効率をさらに高くするためのプログラムがディスクキャッシュプログラムです。

DCACHE2.R はデバイスドライバに割り込んで動作するため、SASI、SCSI、フロッピーディスクなど実際のデバイスに関係なく、キャッシングできます。またコマンドラインから常駐／常駐解除ができるため、そのときのメモリの使用状態にあわせてキャッシュバッファの大きさを変更することができます。

【作者】……………………Arimac

| | |
|---|---|
| NIFTY-Serve | PAF03012 |
| EEL-NET | #87 |
| PC-VAN | VFM10670 |
| J&P-HOT-LINE | JH180342 |

【作者からの言葉】……　DCACHEを作るまではjitte氏作のJET-CACHEというディスクキャッシャを使っていました。その後、XVIを使うようになって、このソフトがSCSIに対応していないことがわかり、せっかくのXVIも実際の使用では従来機に比べて遅く感じられるようになってしまいました。SCSIが採用されてからけっこう時間がたっているので、対応ソフトが出ているのではないか？　と探してはみましたが、結局、見つかりませんでした。調べてみるとIOCSレベルのインタフェースが大幅に変わっていて、それまでの方法が通用しないため

のようです。操作上ではほとんど違いはないので、どこかに違いを吸収するところがあるはず……おお！　それこそはデバイスドライバ！　ここに割り込むことができれば汎用のディスクキャッシャができる！　ということで、割り込むことができるか？　割り込んでも問題はないか？　等を実験して、このプログラムができたわけです。

最初はJET-CACHEを改造しようかと思いましたが、改造するにしても大改造になるようであり、一応市販ソフトだったので公開するには問題がありそうだということで新たに作ることにしました。でも、この手のソフトはOSのバージョンアップで消えてしまう可能性もあり得るので、今のうちが花かもしれないですねぇ(本来はOSでサポートするべきだと思う)。

**【推薦します】**…………　なんといってもDCACHE2.Rは重宝してしています。これを使う前は、XCでソフトを作っていると、ハードディスクのアクセスする音が気になっていたのだけれども、これを使うようになってからは、キャッシュを少し大きめにとっておくだけで全然ハードディスクをアクセスしなくなったので気に入っています。というか、なんとなくディスクアクセスが少ないと、ディスクが長持ちするような気がするから……。それにSX-WINDOWを使って何かを行うときにも、DCACHE2.Rを常駐しておくとアクセス数が減り、快適にウィンドウで遊ぶことができます。絶対お勧めのソフトです。

製作者に感謝いたします。

MAX BBS 869　ハウラ.

**【使用する前に】**………　コマンドラインから実行すると常駐し、動作を開始します。通常はCONFIG.SYSやAUTOEXEC.BATの中からシステムの起動時に実行させます。

CONFIG.SYSから実行する場合はデバイスドライバではありませんので、

```
PROGRAM = DCACHE2 [ドライブ名：[オプション]……]……[オプション]……
```

としてください。常駐すると、キャッシングを行うドライブ名を表示します(Fig.1)。

Fig.1　常駐時の表示例

```
C:>DCACHE2 A:Q B:F -M512
X68k DCACHE v2.01 Copyright 1991～92 Arimac
 キャッシュバッファとして 512 [KB]確保しました。
ドライブ A: B: をキャッシュ対象にします。
```

**【使用法】**………………　すべてのドライブをキャッシングし、なおかつFATを優先的にキャッシングし、キャッシュバッファは1.5Mバイトを確保するように指定する場合。

DCACHE2 -f -m1536[CR]

キャッシング状態を表示させるには、

DCACHE2[CR]

DCACHEの常駐を解除する場合は、

DCACHE2 -r[CR]

とします。

【書式】……………………DCACHE2［ドライブ名：［オプション］……］……［オプション］……

【オプションスイッチ】

| オプションスイッチ | 内　容 | デフォルト |
|---|---|---|
| ドライブ名 | キャッシングするドライブ(A:～Z:)を指定する（複数指定可）<br>ドライブ名のあとには“:”をつけること<br>ドライブ名のあとにq、f、sをつけることによりドライブごとに動作を指定できる<br>q　優先ドライブ<br>f　FAT優先キャッシュ<br>s　新規登録停止 | |
| -h | オプション一覧の表示 | |
| -m数値1[,数値2] | 数値1でキャッシュバッファの容量を指定し、数値2でメインメモリとして最低限残す容量をKバイト単位で指定する。数値のあとに“M”をつけるとMバイト単位になる | |
| -b | キャッシュバッファをメモリの高位アドレスに確保する | 低位アドレスに確保 |
| -q | 指定したドライブを優先的にキャッシュするようにする。ドライブ名のあとに“q”をつけた場合と同じ | |
| -q0 | 指定したドライブを優先的にキャッシュすることをやめる<br>ただし、すでにキャッシュバッファに入っているデータは優先的に扱われる | |
| -f | 指定したドライブのFATおよびルートディレクトリを優先的にキャッシングする | |
| -f0 | 指定したドライブのFATおよびルートディレクトリを優先的にキャッシングすることをやめる<br>ただし、すでにキャッシュバッファに入っているデータは優先的に扱われる | |
| -n | Human68kのバージョンをチェックしない<br>ただし、動作が確認されているのはV2.02 およびV2.03 のみ | チェックする |
| -v | 詳細表示モードにする | |
| -c | 指定したドライブのキャッシュデータを消去する | |

| | | |
|---|---|---|
| -s | 指定したドライブのキャッシングを停止する<br>ただし、すでにバッファにあるデータについては継続される | |
| -g | 指定したドライブのキャッシングを再開する<br>すでにDCACHE2が常駐している場合のみ有効 | |
| -r | 指定したドライブを管理するDCACHE2の常駐を解除する | |

注：

1）オプションの指定は大文字でも小文字でもかまわない。

2）ドライブを指定しない場合は、RAMディスクを除いたすべてのデバイスが対象となる。

3）同一タイプのデバイスを複数接続している場合や、ハードディスクを複数のパーティションに分割して使用している場合、ドライブ名ごとにDCACHE2を常駐させることができるが、その場合、あとから常駐したものから順に解除する。ただし、一度にすべてを解除する場合はこの限りではない（解除できるものから解除していくため）。

【使用法】……………… ドライブA:を優先的にキャッシングし、ドライブB:をFAT優先キャッシュにする場合。

DCACHE2 A:q B:f -m512

すべてのドライブをキャッシングし、なおかつFATを優先的にキャッシングする場合。

DCACHE2 -f -m1536

すでに常駐している場合、オプションをつけずに実行すると、現在のバッファの状態を表示します(Fig.2)。これは、"-s"、"-q"、"-f"などのオプションスイッチをつけ、キャッシングの状態を変更した場合も同様です(Fig.3)。

Fig.2、Fig.3でキャッシュバッファのあとに＊HM＊の表示がありますが、これは"-b"オプションスイッチを指定して、メモリの高位アドレス（アドレスの数字が大きいほう）にキャッシュバッファを確保している場合に表示されます。

このようなオプションスイッチが必要な理由は、X68000の場合、低位側のアドレスのメモリのほうが高位側に比べ速く動作することがあるためです。拡張スロットに増設するタイプのメモリボードの場合、シャープ純正の増設メモリボードでは、メモリをアクセスする場合にはウェイトが入ります。またサードパーティ製のノーウェイトのメモリボードを使用していても、XVI、Compact などで拡張スロットに差された増設メモリは、CPUのクロックスピードに関係なく、10MHzで動作します。したがって、メモリの低位アドレスに大きなサイズのキャッシュ用のバッファを確保すると、アプリケーションプログラムが高位アドレスのメモリに追いやられ、動作速度が遅くなるといったことが起こり得ます。そこで高位アドレスのメモリにバッファを取るように指定することで、それを避けることができます。

ただし、メモリの高位アドレスにキャッシュバッファを確保する場合、低位アドレスに

比べ、プログラムの暴走などによりキャッシュバッファが破壊される可能性が高いと思われるので注意してください。

Fig.2　キャッシュバッファの状態表示

```
C:>DCACHE2
X68k DCACHE v2.01 Copyright 1991～92 Arimac
キャッシュバッファ *HM* 容量 1536 [KB], 使用 135 [KB]
  A: <FD-2HD>    DISK2HD,0       --f      0 [KB]      0 [KB]
  B: <FD-2HD>    DISK2HD,1       --f      0 [KB]      0 [KB]
  C: <HD(SASI)>  HARDDSK,0       --f     15 [KB]     80 [KB]
  D: <HD(SASI)>  HARDDSK,1       --f      0 [KB]      0 [KB]
  E: <HD(SASI)>  HARDDSK,2       --f      0 [KB]      0 [KB]
  F: <HD(SASI)>  HARDDSK,3       --f      3 [KB]     17 [KB]
  G: <HD(SCSI)>  SCHDISK,0       --f      3 [KB]     17 [KB]
  H: <HD(SCSI)>  SCHDISK,1       --f      0 [KB]      0 [KB]
  I: <MO-DISK>   SCHDISK,0       --f      0 [KB]      0 [KB]
  J: <MO-DISK>   SCHDISK,1       --f      0 [KB]      0 [KB]
  K: <MO-DISK>   SCHDISK,2       --f      0 [KB]      0 [KB]
  L: <MO-DISK>   SCHDISK,3       --f      0 [KB]      0 [KB]
```

通常モード

```
                    ┌高位アドレスにバッファを確保すると表示
                    ↓
キャッシュバッファ *HM* 容量 1536 [KB], 使用 135 [KB]
  A: <FD-2HD>    DISK2HD,0       --f      0 [KB]      0 [KB]
  B: <FD-2HD>    DISK2HD,1       --f      0 [KB]      0 [KB]
  C: <HD(SASI)>  HARDDSK,0       --f     15 [KB]     80 [KB]
  L: <MO-DISK>   SCHDISK,3       --f      0 [KB]      0 [KB]
```

- 非優先データ量
- 優先データ量
- 新規登録停止(s)、優先ドライブ(q)、FAT優先キャッシュ(f)
- ユニットNo.
- デバイスドライバ名
- メディアバイトによる表示

Fig.3　詳細表示モード

```
C:>DCASHE2 -v c: -q
X68k DCACHE v2.01 Copyright 1991～92 Arimac
キャッシュバッファ $00086110($003AFBF0) *HM* 容量 1536 [KB], 使用 135 [KB]
  A: <FD-2HD>    DISK2HD($0000F9D0),0    --f     0 [KB]      0 [KB]
  B: <FD-2HD>    DISK2HD($0000F9D0),1    --f     0 [KB]      0 [KB]
  C: <HD(SASI)>  HARDDSK($0000FBEE),0    -qf    15 [KB]     80 [KB]
  D: <HD(SASI)>  HARDDSK($0000FBEE),1    --f     0 [KB]      0 [KB]
  E: <HD(SASI)>  HARDDSK($0000FBEE),2    --f     0 [KB]      0 [KB]
  F: <HD(SASI)>  HARDDSK($0000FBEE),3    --f     3 [KB]     17 [KB]
  G: <HD(SCSI)>  SCHDISK($0007DC84),0    --f     3 [KB]     17 [KB]
  H: <HD(SCSI)>  SCHDISK($0007DC84),1    --f     0 [KB]      0 [KB]
  I: <MO-DISK>   SCHDISK($00080B42),0    --f     0 [KB]      0 [KB]
  J: <MO-DISK>   SCHDISK($00080B42),1    --f     0 [KB]      0 [KB]
  K: <MO-DISK>   SCHDISK($00080B42),2    --f     0 [KB]      0 [KB]
  L: <MO-DISK>   SCHDISK($00080B42),3    --f     0 [KB]      0 [KB]
```

```
詳細モード
                      ┌DCACHEの常駐アドレス
                      ↓          ┌キャッシュバッファのアドレス
                                 ↓
  キャッシュバッファ $00086110($003AFBF0) *HM* 容量 1536 [KB], 使用 135 [KB]
    A: <FD-2HD>    DISK2HD($0000F9D0),0     --f       0 [KB]       0 [KB]
    B: <FD-2HD>    DISK2HD($0000F9D0),1     --f       0 [KB]       0 [KB]
    C: <HD(SASI)>  HARDDSK($0000FBEE),0     -qf      15 [KB]      80 [KB]
    L: <MO-DISK>   SCHDISK($00080B42),3     --f       0 [KB]       0 [KB]

                            ↑
                            └デバイスヘッダのアドレス
```

また、メモリの高位アドレスにバッファを確保するRAMディスクとの相性に注意してください。dcache.lzhアーカイブに含まれているドキュメント(DCACHE2.DOC)に書かれているGRAD.Rの現在のバージョン(V1.25)では、すでに高位アドレスにキャッシュバッファが確保されている場合は常駐しないようになっています。RAMディスクの容量を頻繁に変更する場合も、低位アドレスにキャッシュバッファを確保したほうがよいでしょう。

なお、Fig.3では、“-v”オプションスイッチを使用していますが、この場合、詳細表示モードとなります。

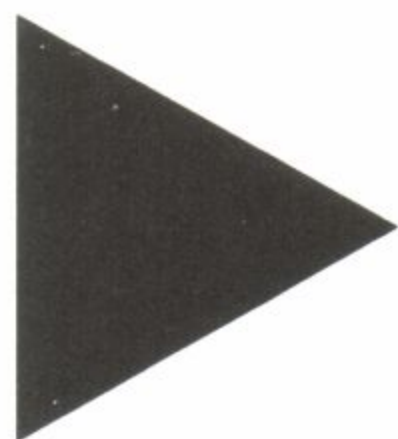
### ●その他

“-s”、“-q”、“-f”オプションを使用する際、別々のドライブに、別々の動作を指定しようとしても、指定されたドライブすべてに、すべてのオプションが作用します。

```
DCACHE2 a: -q b: -f
```

は、

```
DCACHE2 a: b: -f -q
```

と解釈され、ドライブA:、B:ともにFAT優先、優先ドライブ指定となります。

# DE.X

## ドライブ名をデバイスごとに設定するツール

【概要】……………………　Human68kが起動するとき、システムを読み込んだ装置の中でドライブ番号の小さいものから順にA:、B:……とドライブ名がつけられていきます。また、ハードディスクやMOを複数の領域に分割して使用している場合はパーティション番号の小さい領域から順にドライブ名がつけられます。つまり、フロッピーディスクから立ち上げたときと、ハードディスクから立ち上げたときとでは同じドライブ名であっても、実際の装置は変わってくるわけです。

　Human68kに標準で添付されているDRIVE.Xを使えば、ドライブ名を入れ替えて好みの設定にすることができます。しかし、DRIVE.Xでは一度に1組のドライブ名を入れ替えることしかできませんので、装置の種類が増えてくると非常に面倒です。

　DE.Xは、装置の種類ごとに一度にドライブ名を設定でき、また任意の時点で変更することができますので、臨時にデバイスを接続しているようなときにも便利です。

【作者】……………………BAZU　　小田原城下町ネット

【作者からの言葉】……　MOを買った。幸せ。そんなある日、パーティションが違ったり、MOにメディアを入れなかったりして立ち上げると、FDのドライブ名が何になってるかわからない症候群に襲われた。

「このままぢゃいけない」

　そもそもFDのドライブ名なんて固定になってりゃいいのに、なんで勝手に順番がついちゃうのだ？　ええい、MOだってそーだ。ドライブ名はメディアごとに決まった名前から始まってりゃ、問題ないゾ。

「そう思って作りました」

　DE.Xは、SASIだの、SCSIだの、MOだの、鈴なりメディア化してる環境において最大の能力を発揮します。CONFIG.SYSで指定したり、AUTOEXEC.BATで起動するなどにして快適な環境作りにお役立てください。

　ドキュメントにもありますが、DE.XではSCSI番号指定の関係上、システムのSCSIワ

ーク(デバイスドライバ中)を見に行っています。一応、正式なSCSIデバイスドライバにおいてはチェックしましたが、新しいバージョンのもの、特殊なSCSIデバイスドライバについては正常に動作しない恐れがあります(正式バージョンは随時対処するつもり)。

あまりよいソースではございませんが、DE.XにはCのソースリストがついております。DE.Xは、こんな簡単なプログラムですので、改良・改造してよくなるのであれば、どんどん変更して広めてください。

【推薦します】………… はじめてハードディスクを使った感動は忘れられないものですが、ドライブの順番が変わっちゃって使いずらくなったよ～という人も多いはず。標準添付のDRIVE.Xを並べても変えられますが、ここはやはり、これ一発でスマートに。自分好みに仕立てましょう。

MAX BBS　てぃん

【主な使用法】………… フロッピーディスクをA:から、SASIのHDDをC:からドライブ名を設定する場合は、以下のように実行します。

DE /FA /HC[CR]

現在のドライブ名とデバイスの対応関係を表示させる場合は、

DE[CR]

とします。

【書式】………………… DE [オプション……]

【オプションスイッチ】

| オプションスイッチ | 内　容 | デフォルト |
|---|---|---|
| /F? | 標準のドライブ指定 | |
| /A? | 2DD/8セクタFDのドライブ指定 | |
| /B? | 2DD/9セクタFDのドライブ指定 | |
| /C? | 2HD/18セクタFDのドライブ指定 | |
| /D? | 2HD/15セクタFDのドライブ指定 | |
| /H? | SASIのHDDのドライブ指定 | |
| /0?～/7? | SCSIデバイスのドライブ指定(番号はSCSI-IDを示す) | |
| /R? | RAMディスクのドライブ指定 | |
| /Pn | 表示モードの指定(n=0～2)(後述) | /P0 |

注:“?”はドライブ名に置き換えてください。“:”は不要です。

**【使用法】**……………… オプションスイッチをつけずにDE[CR]とだけ実行すると、現在のドライブ名とデバイスの対応関係を表示します。オプションスイッチをつけた場合は、その指定に従って変更された状態を表示します。空いているところは仮想ドライブです。

**●実行例**

標準フロッピーディスクドライブをA:から、SASIのHDDをC:から、ID2、ID4のSCSIデバイスを、それぞれG:、I:から、RAMディスクをZ:に指定する場合は以下のように指定してください(Fig.1)

DE /Fa /Hc /2g /4i /Rz

**Fig.1** 実行例

```
C:>de /Fa /Hc /2g /4i /Rz
DE.x    Ver 0.14 1992/09/07 著作権放棄人 BAZU
A: ２ＨＤ  8 ( 1.2ＭＢ) 標準   ユニット番号････   0
B: ２ＨＤ  8 ( 1.2ＭＢ) 標準   ユニット番号････   1
C: ハードディスク (SASI)      ユニット番号････   0
D: ハードディスク (SASI)      ユニット番号････   1
E: ハードディスク (SASI)      ユニット番号････   2
F: ハードディスク (SASI)      ユニット番号････   3
G: ハードディスク (SCSI-2)    ユニット番号････   0
H: ハードディスク (SCSI-2)    ユニット番号････   1
|
Z: ＲＡＭ ＤＩＳＫ            ユニット番号････   0
```

なお、オプションスイッチで“/P1”を指定した場合は、ドライブの対応状態の表示をしません(Fig2)。

**Fig.2** /P1を指定した場合

```
C:>de /Fa /Hc /2g /4i /Rz /P1
DE.x    Ver 0.14 1992/09/07 著作権放棄人 BAZU
```

また、“/P2”を指定すると、簡易表示になります(Fig.3)。

**Fig.3** /P2を指定した場合

```
C:>de /Fa /Hc /2g /4i /Rz /P2
DE.x    Ver 0.14 1992/09/07 著作権放棄人 BAZU
 A  B  C  D  E  F  G  H  I  J  K  L  M  N  O  P  Q  R  S  T  U  V  W  X  Y  Z
 FD FD HD HD HD HD HD HD MO MO MO MO -  -  -  -  -  -  -  -  -  -  -  -  -  RD
```

**●その他**

基本的に接続されているデバイスについては、すべてドライブ名を指定してください(ただし、SASIのHDDは複数つながれていても、1つのものと見なされます)。指定がないデバイスについては、どこか空いているところに設定されますので注意してください。

また、ドライブ名を変えたときは環境変数のPATHも設定し直す必要が出てくることもあります。特にAUTOEXEC.BATなどでシステムの起動時に実行する場合は、変更を見越してPATHを設定しておかなければなりませんし、その他の環境変数も同様です。

DE.Xで設定したドライブ名とデバイスの対応関係を元に戻すには、Human68kに標準でついてくるDRIVE.Xを使用してください。"DRIVE /D"で起動時の状態に戻ります。このときも環境変数の再設定をすることをお忘れなく。

DE.Xは、SCSIデバイスの情報を得るためにデバイスドライバを直接参照しています。今後システムがバージョンアップし、デバイスドライバが変更された場合、新しいFORMAT.Xでフォーマットされたデバイスを使用すると、異常動作を起こす可能性がありますので注意してください(コラム参照)。ちなみに、DE.X Ver.0.15では、Human68k V2.03付属のFORMAT.X(91-05-15) まで対応しています。

また、同じようなプログラムとして、りる氏(NIFTY-Serve GGA03025)のdrvxchg.xがあります。こちらは、内蔵2HD、FDDおよび増設FDDのドライブ名を変更するのが主な目的のようです。

COLUMN

### SCSIのHDD、MOのデバイスドライバについて

X68000 Super以降のSCSIインタフェースを搭載した機種、もしくはそれ以前の機種で使用するSCSIインタフェースカードにはSCSIDRV.SYSというデバイスドライバがついてきます。しかし、起動用のデバイスがSCSIのHDDの場合（かりに複数のSCSIのHDDがつながっていても）、このデバイスドライバは組み込む必要がありません。

X68000の場合、SCSIのデバイスドライバはメディアに書き込まれており、起動デバイスがSCSIデバイスの場合は、IPLによって起動時に自動的に読み込まれるためです。同時に他のSCSI-IDを検査して、そのIDのデバイスが存在する場合には、そこからもデバイスドライバを読み込みます。フロッピーディスクやSASIのHDDから立ち上げた場合は、SCSIDRV.SYSを使ってIDごとにSCSIのメディアに書かれたデバイスドライバを読み込みます。つまり、SCSI-IDごとに専用のデバイスドライバがシステムの中に用意されるわけです。そのデバイスドライバをメディアに書き込むのがFORMAT.Xなのですが、現在FORMAT.Xのバージョンの違いによって、２種類のデバイスドライバが存在しています。

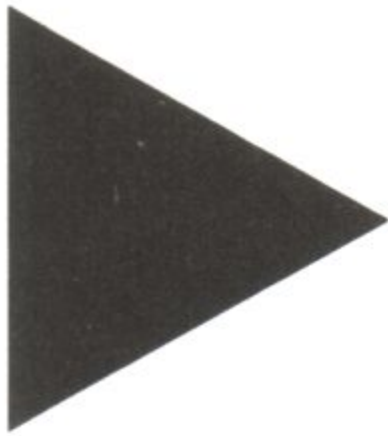
SCSIボード付属／XVI初期およびSX-WINDOW Ver.1.1付属

```
FORMAT      X   95664   90-09-01   12:00:00
HDフォーマット  X   83472   90-09-01   12:00:00
```

XVI後期およびSX-WINDOW Ver.2.0付属

```
FORMAT      X   95976   91-05-15   12:00:00
```

```
HDフォーマット             X     83170     91-05-15   12：00：00
```

の２種類です。91-05-15のタイムスタンプのものは90-09-01のものに比べ、いくつか細かいバグが修正されているようです。
　DE.Xでは、上記２種類のデバイスドライバに対応しています。

# ADDDRV.X　CUTDRV.X
# KEEP.X　TRDRV.X　GRDRV.X
## デバイスドライバ用ユーティリティ

【概要】……………………　ASK68KなどのフロントエンドプロセッサやOPMDRV.Xなどのデバイスドライバは、一度組み込むと、不要になってもそれまで使っていたメモリを解放することができません。また、複数の同じ種類のドライバを切り替えながら使用するといったこともできません。ADDDRV.X、CUTDRV.X、KEEP.X、TRDRV.X、GRDRV.Xは、これらのデバイスドライバを使うときに不便と思われる問題を解消してくれるプログラムです。

【作者】……………………去石　徹　　NIFTY-Serve PEA00566

【作者からの言葉】……　今回は私の作ったプログラムを載せていただき、ありがとうございました。X68000は、初代を発売前に予約注文して買ったんですが、これは趣味のパソコンとして割り切って使うつもりで買ったものでした。実際に、現在も趣味以外の目的には使われていませんが、だからこそ、俗にフリーソフト（当時は、この呼び名もありませんでしたけれども）と呼ばれるプログラムを作れたのかなと思っております。また当時は、欲しいと思うソフトもなかなかなく、ほとんど自作しなければなりませんでした。そういう中で、みんなで使えそうなものをパソコン通信で公開したのが、私がプログラムを発表するようになったきっかけでした。

趣味でやっているプログラム作りでしたから、自分が必要なもの以外は作らないというのが原則で、要望されながら実現しなかった機能なんていうのが結構あります。それでも使っていただいている方が多いというのがうれしいですね。また、そういう方からの貴重なご意見のおかげで、ここで紹介されるような完成されたプログラムにすることができたのだと感謝しております。

ここに紹介されたものの他にも、私が公開したプログラムというのが数本あります。機会があれば使ってみていただけるとうれしいです。また、私の作ったプログラムは、基本的にソース（読みづらいかもしれないけれど）がついておりますので、新しくフリーソフト等を作ろうとしている方々の参考にしていただければと思っております。

【使用する前に】………　付録ディスクに収められたプログラムは２本のアーカイブファイルに分かれています。

```
---rw    31768    89-10-20 02:10:00    drv_210.lzh
---rw     4399    89-11-06 15:14:30    drv211fx.lzh
```

このうち、drv211fx.lzhにはdrv_210.lzhで公表された後に見つかったバグを修正したTRDRV.X、GRDRV.Xが入っていますので、まず最初にdrv_210.lzhを解凍し、その後でdrv211fx.lzhを解凍してください。

## ADDDRV.X

【概要】……………………　**例外はありますが、一般にデバイスドライバはコマンドラインから組み込んだり解除したりすることはできないので、一度組み込むと、不要になってもそれまで使っていたメモリを解放することができません。ADDDRV.Xは、キャラクタ型のデバイスドライバをコマンドラインから追加したり、削除したりすることができるので、メモリを有効に使うことができるようになります。**

【主な使用法】…………　追加登録するデバイスドライバの一覧を書いたファイル（以下、定義ファイルと呼びます）をあらかじめ用意しておき、そのファイル名をパラメータとしてコマンドラインから実行します。

ADDDRV device.def[CR]

【書式】……………………ADDDRV ［オプションスイッチ］［ファイル名］

【オプションスイッチ】

| オプションスイッチ | 内　　容 |
|---|---|
| ファイル名 | 追加登録するデバイスドライバの一覧を書いた定義ファイル |
| q | 追加したデバイスドライバの削除 |

【使用上の注意】………●**このプログラムで登録、削除できるデバイスドライバ**

Human68kのデバイスドライバには、ブロック型のものとキャラクタ型のものの2種類があります。ブロック型デバイスはフロッピーディスクやハードディスクなど、ブロック単位でデータをやりとりするもので、登録された順にA:、B:、……と順に名前がつけられていきます。キャラクタ型デバイスはプリンタやRS-232Cなど、1文字単位でデータをやりとりするもので、それぞれのデバイスに固有の名前がつけられています。たとえば、プリンタならPRNやLPT、コンソール（画面とキーボードを含めた総称）ならばCON、といった具合です。

このうち、ADDDRV.Xで扱えるデバイスドライバはキャラクタ型のものだけです。ただし、キャラクタ型のデバイスドライバとしてCONFIG.SYSから組み込めるものでも、ADDDRV.Xが管理する以外のベクタを変更するものもあり（HIOCS.X、condrv.sys、TwentyOne.xなど）、プログラムによってはADDDRV.Xを使用して登録、削除すると、

誤動作する場合がありますので、注意してください。

●定義ファイルの内容

定義ファイルの中身は、登録するデバイスドライバのファイル名とオプションスイッチを並べたものです。つまり、CONFIG.SYSから登録する際のDEVICE=という部分を除いたものになります。

たとえばCONFIG.SYSの中で、

DEVICE = ¥sys¥ASK68K.SYS G:¥DIC¥X68K_M.DIC G:¥DIC¥X68K_S.DIC ¥ETC¥ENV.ASK

としていた場合は、

¥sys¥ASK68K.SYS G:¥DIC¥X68K_M.DIC G:¥DIC¥X68K_S.DIC ¥ETC¥ENV.ASK

となります。

また、次のように環境変数を使ってデバイスドライバのパラメータを変更することもできます。

¥sys¥PRNDRV1.SYS %PRNDRV%

この場合、ADDDRV.Xを実行する前に環境変数PRNDRVに必要なパラメータをセットしておけば、%PRNDRV%の部分が環境変数の内容に置き換えられますので、組み込み時にパラメータを変更することができます。

## CUTDRV.X

【概要】…………………… CUTDRV.Xは、ADDDRV.Xによって最後に登録されたデバイスドライバを一時的に切り離します。再度CUTDRV.Xを実行すれば、再びそのデバイスドライバを登録する前の状態に戻ります。デバイスドライバを切り離すときは、デバイスドライバが使用しているメモリを解放せずに、変更したベクタを元に戻すだけなので、ADDDRV.Xで登録、解除を繰り返すより素早く行えます。また、2つのフロントエンドプロセッサを切り替えながら使用する場合にも使えますが、その分メモリを多く使うので注意してください。

【使用法】………………… ADDDRV.Xでデバイスドライバを登録したあと、CUTDRV.Xを実行すれば、そのデ

バイスドライバが切り離されます。

同じ種類の2つのデバイスドライバを切り替える場合（たとえば、ASKとFIXERを切り替える場合）は、各々別々の定義ファイルを用意してADDDRV.Xで両方登録してください。そのあと、CUTDRV.Xを実行すれば、最後に登録されたデバイスドライバが切り離され、その前に登録した1つ目のデバイスドライバが有効になります。もう1度CUTDRV.Xを実行すれば、2つ目の（最後に登録した）デバイスドライバが再度有効になります。ただし、あとで紹介するTRDRV.X、GRDRV.Xで登録したデバイスドライバは切り離すことができませんので注意してください。

例）

ADDDRV ask.def[CR]

ADDDRV fixer.def[CR]　　（ask.defとfixer.defは各々の定義ファイル）

と実行すると、ASKとFIXERの両方が登録されますが、あとに登録したFIXERがフロントエンドプロセッサとして有効になります。そのあと、

CUTDRV[CR]

と実行すると、一時的にFIXERが切り離され、今度はASKが有効になります。再度CUTDRV.Xを実行すると、再びFIXERが有効になります（Fig.1）。

**Fig.1** CUTDRVによるデバイスドライバの切り離し

```
C:>adddrv ask.def

adddrv.x  Ver2.10  by T.Saruishi  89/10/20


日本語フロントプロセッサ ＡＳＫ６８Ｋ for X68000 version 2.01
Copyright 1987,88,89 SHARP Corp./ACCESS CO.,LTD.

C:>adddrv fixer.def

adddrv.x  Ver2.10  by T.Saruishi  89/10/20


日本語入力プロセッサ　ＦＩＸＥＲ６８Ｋ　Ver. 1.20
 Copyright (C) 1986, 92 CITYSOFT Corp.

C:>cutdrv

cutdrv.x  Ver2.10 ( for adddrv.x Ver2.10 )  by T.Saruishi  89/10/20

 デバイスドライバを一時的に切り離しました
```

これは、

ADDDRV ask.def[CR]　(ASKを登録する)
ADDDRV q[CR]　(ASKを削除する)
ADDDRV fixer.def[CR]　(FIXERを登録する)

とするのとほぼ同じです。CUTDRV.Xを利用するほうが再ロードが不要な分だけ速く切り替えられます。しかし、その分、メモリがよけいに必要になります。

【オプションスイッチ】オプションスイッチはありません。

## KEEP.X

【概要】……………………**常駐プログラムを強制的に終了させるプログラムです。常駐型のプログラムを実行する前にKEEP.Xを実行することで、OPMDRV.Xなど、一度常駐するとメモリを解放できないプログラムでも解放することができるようになります。**

【使用法】………………　常駐プログラムを実行する前に、

KEEP[CR]

と、オプションなしで実行しておき、そのあと、常駐プログラムを実行します。再度オプションなしで実行すると、KEEP.Xおよび常駐プログラムが使用していたメモリが解放されます。

【書式】……………………KEEP［オプションスイッチ］

【オプションスイッチ】

| オプションスイッチ | 内　　容 | デフォルト |
|---|---|---|
| S | すでにKEEP.Xが常駐していてさらに常駐させる | |
| Q | 複数のKEEP.Xをすべて終了させる場合 | |

　オプションスイッチがない場合、KEEP.Xが常駐していないときは常駐し、すでに常駐している場合は最後に実行されたKEEP.Xが終了します。

【使用法】………………**1．一般的な使用例**

　まず、コマンドラインからオプションなしで、KEEP[CR]と実行し、たとえばその後、OPMDRV.Xをコマンドラインから常駐させます。そのあと、再度KEEP[CR]と実行す

ると、OPMDRV.Xが切り離され、OPMDRV.Xが使用していたメモリも解放されます(Fig.2-a)。

**Fig.2-a**　KEEP.Xによるデバイスドライバの切り離し

```
G:¥usr>keep

keep.x  Ver2.10   by T.Saruishi   89/10/20

  ベクタを保存しました

G:¥usr>opmdrv

ＦＭ音源 DRIVER for X68000 version 1.02
OPMのファイル名でＦＭ音源に出力可能です

G:¥usr¥y¥books¥genko¥up>ps
PID    PPID   SIZ    SIZ(K)  STAT NAME
007D50 000000 07E3B0  504.92 SUPR HUMAN.SYS
086100 007D50 000F1A    3.77 KEEP tmsio.x
087020 007D50 18B2E2 1580.72 KEEP dcache2.r
0876F0 007D50 0097CE   37.95 USER COMMAND.X
090ED0 0876F0 000FB4    3.92 KEEP keep.x
091E90 0876F0 015482   85.12 KEEP opmdrv.x
0A7320 0876F0 3088C0 3106.18 SUPR ps.r
53A7F4        3088C0 3106.18
G:¥usr>keep

keep.x  Ver2.10   by T.Saruishi   89/10/20

  ベクタを復帰しました。

G:¥usr¥y¥books¥genko¥up>ps
PID    PPID   SIZ    SIZ(K)  STAT NAME
007D50 000000 07E3B0  504.92 SUPR HUMAN.SYS
086100 007D50 000F1A    3.77 KEEP tmsio.x
087020 007D50 18B2E2 1580.72 KEEP dcache2.r
0876F0 007D50 0097CE   37.95 USER COMMAND.X
090ED0 0876F0 31ED10 3195.26 SUPR ps.r
53A7F4        31ED10 3195.26
```

## 2．KEEP.Xを複数常駐させる場合

使用例の１と同様に、OPMDRV.Xを常駐させたあと、“S”オプションスイッチをつけてKEEP.Xを実行します。その後、tokei.x(CRT上にアナログ時計を表示するフリーソフトウェアです。tokei.x自身で常駐解除できますが、例として出しました）を実行します。そして、再度KEEP.Xを実行すると、tokei.xのみが終了します(Fig.2-b)。

Fig2-b keep.xの複数常駐

```
G:¥usr>keep

keep.x  Ver2.10   by T.Saruishi   89/10/20

  ベクタを保存しました

G:¥usr¥y¥books¥genko¥up>opmdrv

ＦＭ音源 DRIVER for X68000 version 1.02
OPMのファイル名でＦＭ音源に出力可能です
G:¥usr>keep S

keep.x  Ver2.10   by T.Saruishi   89/10/20

  ベクタを保存しました

G:¥usr>tokei
G:¥usr>ps
PID    PPID   SIZ    SIZ(K)  STAT NAME
007D50 000000 07E3B0  504.92 SUPR HUMAN.SYS
086100 007D50 000F1A    3.77 KEEP tmsio.x
087020 007D50 18B2E2 1580.72 KEEP dcache2.r
0876F0 007D50 0097CE   37.95 USER COMMAND.X
090ED0 0876F0 000FB4    3.92 KEEP keep.x
091E90 0876F0 015482   85.12 KEEP opmdrv.x
0A7320 0876F0 000FB4    3.92 KEEP keep.x
0A82E0 0876F0 000676    1.61 KEEP tokei.x
0A8960 0876F0 307280 3100.62 SUPR ps.r
53A7F4        307280 3100.62
G:¥usr>keep

keep.x  Ver2.10   by T.Saruishi   89/10/20

  ベクタを復帰しました。

G:¥usr¥y¥books¥genko¥up>ps
PID    PPID   SIZ    SIZ(K)  STAT NAME
007D50 000000 07E3B0  504.92 SUPR HUMAN.SYS
086100 007D50 000F1A    3.77 KEEP tmsio.x
087020 007D50 18B2E2 1580.72 KEEP dcache2.r
0876F0 007D50 0097CE   37.95 USER COMMAND.X
090ED0 0876F0 000FB4    3.92 KEEP keep.x
091E90 0876F0 015482   85.12 KEEP opmdrv.x
0A7320 0876F0 3088C0 3106.18 SUPR ps.r
53A7F4        3088C0 3106.18
```

## 3. KEEP.Xを複数常駐させ、すべて常駐解除する場合

使用例の２.同様に、OPMDRV.Xとtokei.xを常駐させます。その後、KEEP.Xに“Q”オプションスイッチをつけて実行すると、両プログラムが終了します(Fig.2-c)。

Fig2-c　複数デバイスドライバの常駐解除

```
G:¥usr>keep

keep.x  Ver2.10   by T.Saruishi   89/10/20

  ベクタを保存しました

G:¥usr>opmdrv

ＦＭ音源 DRIVER for X68000 version 1.02
OPMのファイル名でＦＭ音源に出力可能です
G:¥usr>keep S

keep.x  Ver2.10   by T.Saruishi   89/10/20

  ベクタを保存しました

G:¥usr>tokei
G:¥usr>keep Q

keep.x  Ver2.10   by T.Saruishi   89/10/20

  一括で  ベクタを復帰しました。

G:¥usr¥y¥books¥genko¥up>ps
PID    PPID   SIZ    SIZ(K)  STAT NAME
007D50 000000 07E3B0  504.92 SUPR HUMAN.SYS
086100 007D50 000F1A    3.77 KEEP tmsio.x
087020 007D50 18B2E2 1580.72 KEEP dcache2.r
0876F0 007D50 0097CE   37.95 USER COMMAND.X
090ED0 0876F0 31ED10 3195.26 SUPR ps.r
53A7F4        31ED10 3195.26
```

## TRDRV.X　GRDRV.X

**【概要】**……………………　デバイスドライバはメインメモリに置かれます。メインメモリが少ないとき、日本語フロントエンドプロセッサのような巨大なプログラムを使うとメモリが足りなくなり、アプリケーションが使えなくなるといったことが起こります。このプログラムTRDRV.X、GRDRV.Xは、ふだんあまり使われていないテキストVRAM、グラフィックVRAMにデバイスドライバの本体を置き、メインメモリの空き領域を増やすためのプログラムです。TRDRV.XはテキストVRAMに、GRDRV.XはグラフィックVRAMにデバイスドライバを置くプログラムです。

**【主な使用法】**…………　ADDDRV.Xと同様に、定義ファイルをパラメータとして渡します。ただし、各々1度ずつしか登録できません。

**【書式】**……………………TRDRV［ファイル名］
GRDRV［ファイル名］

**【オプションスイッチ】**

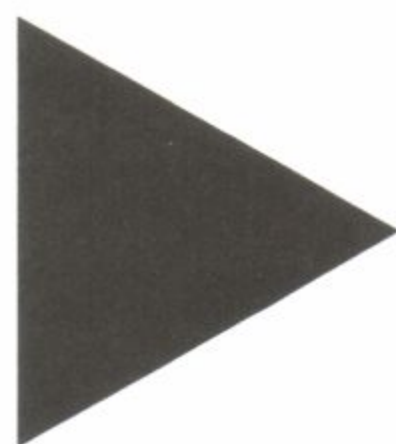
| オプションスイッチ | 内　　容 |
|---|---|
| ファイル名 | 追加登録するデバイスドライバの一覧を書いた定義ファイル名 |

定義ファイルを指定しない場合は、組み込みを解除します。

**【使用法】**

ADDDRV.Xと同様に、定義ファイルを使用してデバイスドライバを登録します。ただし、複数回登録することはできません。組み込みを解除する場合は定義ファイルを指定せずに、TRDRV[CR]、GRDRV[CR]と実行して解除してください。

Human68kのIOCSコールにテキストVRAM、グラフィックVRAMの使用状態(未使用もしくは使用中など)を設定、読み出しするTGUSEMD(IOCSコール番号 $0E)があります。TRDRV.X、GRDRV.Xでも、これを使用してVRAMに置いたデバイスドライバの破壊を防いでいますが、アプリケーションプログラムによっては、このコールを使用せず、無条件にVRAMを破壊するものもありますので、注意して使ってください。

また、TRDRV.X、GRDRV.Xは他のプログラムからではなく、TRDRV.X、GRDRV.X自身で解除してください。KEEP.Xを使って解除すると、VRAMが使用中のままになってしまい、VRAMを使う他のプログラムが動作しないことがあります。

# C/DINIT.SYS

## 起動時に複数のCONFIG.SYSを選択することができるデバイスドライバ

【概要】……………………　たとえば、あなたがプログラムを実行しようとしたとき、「メモリが足りません」というエラーメッセージが出て困ったことはありませんか？　最近のプログラムはどれも機能が豊富な分、大量のメモリを必要とします。1Mバイトではとても足りず、2Mバイトでも快適な環境を得ようと思うと十分ではありません。パソコン通信で手に入れたフリーソフトを使い始めたり、SX-WINDOWを使おうとすると4Mバイトでも足りないという状況になっています。

こんなとき、どうしても目的のプログラムを実行したければ、ふだん使用している快適な環境を我慢し、フリーエリアを確保するようにしなければなりません。たとえば、便利だけれど、今回の目的のプログラムでは使用されないデバイスドライバの登録をやめたり、ディスクのバッファを削ったりして、少しでもメモリを広くしたいと思いませんか？

また、パソコン通信などで手に入れたデバイスドライバなどをちょっと試してみたい場合や、自作のデバッグ中のプログラムを組み込んでみる場合、あなたはどうしていますか？　いきなりCONFIG.SYSに書き加えてしまうのはかなり勇気がいるものです。プログラムや、その使い方に問題があった場合は、最悪の場合、システムが立ち上がらなくなってしまいます。

こんなとき、試しに作ったCONFIG.SYS（に相当するもの）で一時的にシステムを起動し、問題があれば元の状態で再起動できるようにしておければ安心です。

また、コマンドラインから直接登録したり、解除したりできる、“OPMDRV3.X”のような“＊.X”形式のデバイスドライバにしても、システムを構成する重要性を考えれば、暴走などで簡単に書き換わってしまう「ユーザ領域」に置いておくことに不安を抱く方もおられるでしょう（CONFIG.SYS上で登録されたデバイスドライバは、「スーパーバイザ領域」に置かれるので、スーパーバイザ・モードに処理を移す手続きがないとエラーメッセージを出して処理を中断します。そのため、暴走によるメモリの書き換えをかなり防止することができます）。

“CINIT.SYS”、“DINIT.SYS”は、そんなあなたの願いを実現するツールです。CONFIG.SYSのデバイス行の先頭で“CINIT.SYS”または“DINIT.SYS”をデバイスドライバとして登録しておけば、あらかじめ設定した環境の中から目的にあった環境を最大9つまで選択して起動することができます。

“CINIT.SYS”はCONFIG.SYSで設定できるすべての環境を変更することが可能です。たとえば、BELLやBUFFERS、PROCESSなどを変更したい場合に使用

します。BELLは好きなBEEP音を選択できるものですが、BUFFERSとともに、これらのシステムが占有するメモリを減らしたい場合に効果があります。PROCESSは、ユーザが“Timer-D”を使用したい場合、解除しておかなければなりません。

一方、“DINIT.SYS”は、CONFIG.SYS中のDEVICE行、ENVSET行、SHELL行、PROGRAM行の設定を変更することができます。単純にフリーエリアを増加させたい場合は、DINIT.SYSを使ってデバイスドライバを削るのが一番効果がありますし、環境の変更も大部分はデバイスドライバの変更によって実現できます。

なお、CONFIG.SYSの各設定、名称、機能はHuman68kのユーザーズマニュアルを参照してください。

【作者】……………………HYUN　NIFTY-Serve　NBC00117
アスキーネット　pcs30291
Net Noise Race　#50

【作者からの言葉】……　とある草の根ネットでの会話。

某X68ユーザ：X68000にもINIT.SYSみたいのないかぁ？
私　：INIT.SYSってなあに？
某X68ユーザ：知らんの？　ほれ、あのデバドラなんかを組み換えるやつだよ、PC-98用の。
私　：へっ!?

よく聞けば、なにやらCONFIG.SYSの内容を起動時に変えてしまうという、PC-98用の怪しげなドライバらしい。「世界は１つ、CONFIGは１つ」と思っていた私にとって、これは驚きでした。それから３日間は妻子を顧みず、寝食を惜しんで（かなり大げさですが……）Human68kの解析とコーディングに熱中する羽目になったのです。

そんなこんなで(どんなだ？)、デバイスドライバの組み換えを中心としたDINIT.SYSと、CONFIG.SYSの記述内容をほぼ完全（Complete）に組み換えるCINIT.SYSの２本が完成したというわけです。

さて、これはお願いですが、できればDINIT.SYSをお使いになってください。CINIT.SYSは起動に時間がかかりますし、なんといっても人サマのCONFIG.SYSを(一時的にではありますが)リネームするという、大それたことをやってのけますから。「その瞬間に停電にでもなったら……あぁ、公開するんじゃなかった」と、当時は思ったものです。今は大分厚顔になったので、平気ですが。

今もいろいろ危ないソフトを手掛けていますので、折りを見て公開していきたいと思っ

ています。そのときはこわがらずにつきあってくださいね。では。

## CINIT.SYS

まず、CINIT.SYSのほうから設定してみましょう。

CINIT.SYSは、CONFIG.SYSと同一の書式で書かれた設定ファイルを複数用意しておき（最大9つまで）、起動時および再起動時にそのうちの1つを選択して、その設定ファイルの環境でX68000を使用するドライバです。CINIT.SYSは、あとで説明するDINIT.SYSと異なり、CONFIG.SYSで設定できる項目はすべて設定ファイルに記述可能です。

**注意**

CINIT.SYSは、起動時にディスクに対して書き込みを行います（詳しい動作の仕組みは、ドキュメント中に書かれていますので、そちらを参照してください。その仕組みに「なるほど」と感心することでしょう）。このため、CINIT.SYSを使用するディスクは、ライトプロテクトシールを貼るなどの書き込み禁止処置をしないでください。

**【使用法】**

### ●インストール

まず、あなたのディスク上の適当なディレクトリにCINIT.SYSをコピーしてください。他のデバイスドライバと同じディレクトリに入れておくとよいでしょう。筆者の場合は、デバイスドライバは“¥sys”ディレクトリにまとめてありますので、CINIT.SYSもそこに入れてあります。

### ●設定ファイルを作る

設定ファイルはまとめて1つのディレクトリに入れておかなければならないので、適当な名前のディレクトリを作ってください。筆者の場合は、“¥SETUP”というディレクトリにまとめています。設定ファイルは通常のCONFIG.SYSとまったく同じ書式なので、Human68kのユーザーズマニュアルを参照し、自分が選択して使用していきたい環境を作っておきましょう。CINIT.SYSを使えば最大9つまで設定ファイルを用意できますが、ファイル名の拡張子は必ず“＊.SYS”にしてください。筆者は、CINIT.SYSをBUFFERSの容量やコモン、シェアの設定などに使用しています。

### ●CINIT.SYSの設定

さて、準備ができたら、今まで使用していたCONFIG.SYSを、念のため別のディスクにコピーするか、ハードディスクをお使いならば別のパーティションにコピーするなりして保存しておいてください。最低でも安全のため、リネームしておくことをお勧めします。CINIT.SYSはCONFIG.SYSを書き換えて動作します。用意ができたら、今まで使用していたCONFIG.SYSをエディタで読み込んでください。デバイスドライバを設定しているDEVICE行の最初にCINIT.SYSを設定します。

DEVICE = [パス名]¥CINIT.SYS [オプションスイッチ] 設定ファイルのあるディレクトリ

パス名は、CINIT.SYSがあるディレクトリを指定します。オプションスイッチは、起動時に設定ファイル選択メニューを表示させるトリガーキー[注]を指定するためにあります。オプションスイッチを指定すれば、指定されたキーが起動直後に押されている場合のみ、選択メニューが表示されます。もし指定したキーを押さずにそのまま起動させると、今まで使用していたCONFIG.SYSの環境で起動します。

また、なにもオプションスイッチを指定しなければ、起動のたびに毎回選択メニューを表示します。

注：あることが起こるきっかけになるもの。ここでは、設定メニューを表示させるCINIT.SYSの機能を働かせるために押す、キーボード上のキーのことを指す。

**【オプションスイッチ】** CINITのオプションスイッチには、以下のものがあります

| オプションスイッチ | 内容 |
|---|---|
| /S | 起動時に[SHIFT]キーが押されていたら、選択メニューを表示する |
| /C | 起動時に[CTRL]キーが押されていたら、選択メニューを表示する |
| /K | 起動時に[かな]キーが押されていたら、選択メニューを表示する |
| /R | 起動時に[ローマ字]キーが押されていたら、選択メニューを表示する |
| /O | 起動時に[コード入力]キーが押されていたら、選択メニューを表示する |
| /A | 起動時に[CAPS]キーが押されていたら、選択メニューを表示する |
| /I | 起動時に[INS]キーが押されていたら、選択メニューを表示する |
| /H | 起動時に[ひらがな]キーが押されていたら、選択メニューを表示する |
| /Z | 起動時に[全角]キーが押されていたら、選択メニューを表示する |

オプションスイッチのうちLEDキー([CAPS]キーなどのようにキートップにランプの内蔵されたキー）は、起動直後にランプが点灯している状態にしておけば選択メニューが表示されますが、[SHIFT]キーや[CTRL]キーなどのLEDのついていないキーは、選択メニューが表示されるまで、キーを押しっぱなしにしておいてください。LEDキーは、選択メニューが表示されたあと、起動時にSRAMに記憶されている状態に戻ります。

なお、オプションは、“/KHZ”のように複数の指定を連続して表記したり、“/K /H /Z”のように分けて表記することもできます。なお、大文字・小文字の区別はありません。

### ●設定ファイルのあるディレクトリ

設定ファイルのあるディレクトリには、最初に用意した設定ファイルを集めたディレクトリのパス名を書いてください。

### ●CINIT.SYSの起動

それでは、リセットしてシステムを再起動させてみましょう。正しく設定され、システ

ムにCINIT.SYSが組み込まれていれば、指定したオプションスイッチのキーを押しておくだけで、設定ファイルの選択を要求してきます。

Fig.1　設定ファイル選択画面

```
Human68k for X68000 version 2.03
Copyright 1987,88,89,90,91,92 SHARP/Hudson

<< DINIT.SYS        Version 1.03   by HYUN >>

  [Available file(s) in '¥CONFIG¥SETUP¥']

    1. MAX_MEM.SYS
    2. FEP_ASK.SYS
    3. FEP_FIX.SYS
    4. NEC_PRN.SYS
    5. EPJ_PRN.SYS
   (Only <return> to continue this 'CONFIG.SYS')

  Choose the number :
```

うまくメニューが出なかった場合はエラーメッセージをよく読んで解決してください。

CINIT.SYSのあるパス名の指定間違いや、オプションスイッチの指定間違い、設定ファイルのあるディレクトリを間違えている可能性があります。ふだんあまりCONFIG.SYSなどをいじることがないという人の場合は、簡単な綴りミスや入力ミスもありそうです。

うまく、メニューが表示されたら、メニュー中のファイル名から目的の設定を探し、先頭に付加されている番号を入力してください。そのまま[CR]キーを入力すると、メニューはキャンセルされ、CONFIG.SYSの起動を再開します。

番号を入力すると、ディスクにアクセスしたあと、リセットがかかります。このため、CINIT.SYSは起動に若干時間がかかります。しかし、起動後にエディタを立ち上げてCONFIG.SYSを毎回書き直すよりも楽に、かつほぼ自動的に環境の変更をすることができます。

## DINIT.SYS

DINIT.SYSの用途は、CINIT.SYSと同様、ユーザの使用する環境を起動時に変更できるデバイスドライバです。CINIT.SYSとの違いは概要でも挙げたように、CINIT.SYSがCONFIG.SYSファイル中のすべての環境設定を変更することが可能なのに対し

て、DINIT.SYSはDEVICE行の変更を主な目的として用意されています。

あなたが今まで使用していたCONFIG.SYSのうち、DINIT.SYSによって起動時に選択、変更が可能なものはDEVICE行、ENVSET行、SHELL行、PROGRAM行のみとなります（ただし、DEVICE行以外の変更機能は、DINIT.SYSでは「拡張機能」と呼ばれています)。機能に制限こそあるものの、ユーザが変更したい環境はほとんどの場合、この4点の変更で実現できるはずです。筆者の環境の変更も“PRN”というデバイス名が重なってしまうため、使用できない“PRNDRV1.SYS”と“PRNDRV2.SYS”の選択や、“ASK68K.SYS”と“FIXER.SYS”の日本語FEPの選択、RAMディスクの容量変更などのデバイスドライバの選択がほとんどです。

特にメモリの不足を感じている方は、「ワープロ用に辞書をRAMディスクに持った環境」や「プログラム開発のために日本語FEPをはずし、RAMディスクにコンパイラを用意した環境」など、そのときの用途に応じて無駄のない環境を用意したいものです。ですから、不要なデバイスドライバがはずせれば、フリーエリア増加に大変効果があります。

また、このような制限のかわりにDINIT.SYSはCINIT.SYSのようなリセット、再起動を行いません。このため、DINIT.SYSを使えば高速な起動が可能です。また、ディスクに書き込む処理がないため安全ですし、ディスクを書き込み禁止にしておいても大丈夫な点もメリットといえるかもしれません。

**【使用法】**

基本的にCINIT.SYSと同様です。DINIT.SYSを適当なディレクトリに入れてください。DINIT.SYSも設定ファイルを最大9つまでで、拡張子が“.SYS”ならばファイル名は自由につけられます。わかりやすいファイル名にすればよいでしょう。設定ファイルの書式は、通常のCONFIG.SYSと同様ですが、基本的にはDEVICE行のみ、設定が可能です(あとで述べる「拡張機能」を使用するオプションに指定すれば、ENVSET行、SHELL行、PROGRAM行も設定可能になります)。

DINIT.SYSの書式は、

DEVICE=［パス名］¥DINIT.SYS ［オプションスイッチ］設定ファイルのあるディレクトリ

です。

例)

```
DEVICE = ¥SYS¥DINIT.SYS /XC ¥SETUP
```

DINIT.SYSの設定、起動はCINIT.SYSとほぼ同様です。デバイスドライバを指定するDEVICE行の最初に上の例のような1行を追加してください。

**【オプションスイッチ】**　基本的にCINIT.SYSと同様です。「起動時にどのキーが押されたら（あるいは、どのLEDキーが点灯していたら）設定メニューを表示するか」をオプションスイッチで指定します。注意事項もCINIT.SYSと同様です。ただ1つ、CINIT.SYSと異なるオプションスイッチがあります。それが先述の「拡張機能」の「あり／なし」を指定するオプションスイッチです。

| オプションスイッチ | 内　容 | デフォルト |
|---|---|---|
| /X | 拡張機能を使用しない<br>DEVICE行のみ環境を選択した設定ファイルに従い、あとは通常使用しているCONFIG.SYSの環境を使用する | 拡張機能を使用する<br>DEVICE行、ENVSET行、SHELL行、PROGRAM行の環境を選択した設定ファイルに従って起動する |

起動もCINIT.SYSと同様ですが、リセット、再起動がない分、手軽に環境を変更できます。メモリの不足をデバイスドライバの登録で調整したいとお考えの方、起動時にデバイスドライバの変更を選択したいとお考えの方はDINIT.SYSのほうをお勧めします。たとえば、メモリが足りないときに日本語FEPやプリンタを使用しないプログラムを実行するのならば、これらのデバイスドライバを常駐させておく必要はありません。ワープロを使用するとき、標準の日本語FEP、ASK68KとFIXER.SYSを使い分けなければならないこともあります（残念ながら、X68000のワープロソフトはASK以外の日本語FEPで使用されることを想定していないようで、FIXER.SYSでは完全には使用できないものもあります）。こういった場合にDINIT.SYSは活躍してくれるはずです。

**【INICHK.X】**………　起動時にX68000の環境を設定するファイルはCONFIG.SYSだけではありません。AUTOEXEC.BATもユーザの環境構築には重要な役割を果たします。CDINIT.LZHにアーカイブされているINICHK.Xは、ユーザがC/DINIT.SYSの設定ファイルのうち、どれを用いて起動したのかを調べる外部コマンドです。AUTOEXEC.BAT中で使用することで、起動時の設定ファイルに応じて処理を振り分けることが可能になります。詳しくはINICHK.Xのドキュメントを読んで使用してください。

ここでは、使用法と使用例のみを挙げておきます。

**【使用法】**…………………　**書式：INICHK.X**［設定ファイル名］

コマンドラインよりINICHK.X［CR］と入力すると、画面に起動時に使用した設定ファイル名が表示されます。たとえば、通常に起動したときは“CONFIG.SYS”と表示されますし、あなたが“TEST.SYS”という設定ファイルを用意していて、CINIT.SYSまたはDINIT.SYSで“TEST.SYS”を選択して起動した場合は“TEST.SYS”と表示されます。

例）
INICHK[CR]

CONFIG.SYS

　また、後ろにあなたが起動時にCINIT.SYSまたはDINIT.SYSで使用するために用意した設定ファイル名を書き並べておくと、書き並べられた設定ファイル名中、何番目のものが起動時に使用されたかを、エラーコードに返します（エラーコードは、外部コマンドが終了したとき、シェルに外部コマンドの終了状態を知らせるなどの役割のため用意されています。バッチファイル中では、if文のERRORLEVELもしくはEXITCODEを使って知ることができます。詳細は、Human68kユーザーズマニュアルを参照してください）。

例）
INICHK CONFIG.SYS MAXMEM.SYS FIX.SYS ASK.SYS

1　　　←EXITCODEの値です。

　この場合、通常起動に使用しているCONFIG.SYS以外に、メモリをできるだけ確保するようにした設定ファイルMAXMEM.SYS、日本語FEPにFIXER.SYSを使用したFIX.SYSと、同じくASK68K.SYSを使用したASK.SYSというファイル名の設定ファイルを用意しています。
　それぞれ内容は、

```
MAXMEM.SYSの参考例。 ----- メモリをできるだけ確保する設定ファイルの例
DEVICE = ¥SYS¥FLOAT2.X
DEVICE = ¥SYS¥IOCS.X

FIX.SYSの参考例。    ----- 日本語FEPにFIXERを使用する設定ファイルの例
DEVICE = ¥SYS¥FLOAT2.X
DEVICE = ¥SYS¥IOCS.X
DEVICE = ¥SYS¥FIXER.SYS ¥ENV¥FEP¥ENV1.FIX

ASK.SYSの参考例。    ----- 日本語FEPにASK68K.SYSを使用する設定ファイルの例
DEVICE = ¥SYS¥FLOAT2.X
DEVICE = ¥SYS¥IOCS.X
DEVICE = ¥SYS¥ASK68K.SYS ¥DIC¥X68K_M.DIC ¥DIC¥X68K_S.DIC ¥ENV¥FEP¥ENV1.ASK
```

となります。
　なお、最後に筆者が普段使用しているCONFIG.SYSを示しておきます。

```
FILES     = 20
BUFFERS   = 10 1024
VERIFY    = OFF
BREAK     = ON
LASTDRIVE = Z:
*TITLE     = ¥CONFIG¥TITLE¥METAL.TTL
```

```
BELL       = ¥CONFIG¥BEEPS¥STBR.PCM
KEY        = ¥CONFIG¥KEY.SYS
USKCG      = ¥CONFIG¥USKCG.SYS
COMMON     = 20
SHARE      = 20 20
PROCESS    = 32 10 100
DEVICE     = ¥SYS¥DINIT.SYS /C ¥CONFIG¥SETUP
DEVICE     = ¥SYS¥PCMDRVx.SYS
DEVICE     = ¥SYS¥PRNDRV1.SYS
DEVICE     = ¥SYS¥FLOAT3p.X
DEVICE     = ¥SYS¥HIOCS.X /J /MS3
DEVICE     = ¥SYS¥INTSWDRV.X
DEVICE     = ¥SYS¥TWENTYONE.X /S
DEVICE     = ¥SYS¥HISTORY2.X /AR /D¥ENV¥HIS¥ /SH2,8,4 /I
DEVICE     = ¥DEV¥FLEXDISK.SYS #M640 /A /C /P
SHELL      = ¥SHELL¥COMMAND.X /P /E:5
```

INICHK.Xを使って処理を分けるAUTOEXEC.BATの書き方について簡単な例を挙げてみます。この場合、日本語FEPの違いはAUTOEXEC.BATには関係ないので、メモリ確保の“MAXMEM.SYS”だけを処理しています。

```
ECHO OFF
¥DEV¥9SCDRV.X                              ----- 9セクタのフロッピーディス
                                                  クドライバ
¥COM¥BAT_COM¥SETPATH.X ¥ENV¥SETPATH.DEF    ----- パスを設定している
¥COM¥DISK_COM¥DE /FA /OC /RX               ----- ドライブナンバーを揃えてい
                                                  る
C:
INICHK CONFIG.SYS MAXMEM.SYS               ----- ここでは、MAXMEM.SYSで起動
                                                  したかどうを調べている
if exitcode 2 goto MAXMEM
SET temp=X:
SET HOME=F:¥USER
¥COM¥CNS_COM¥RSHIFT /S /D /E
SET TMNCNF=E:¥TOOLS¥TMN¥TMN.CNF
PROMPT $E[41m=====$E[m □ □□□□□ $E[41m======================$E[36m $D $T $E
[41m============================$E[m$E[36 m□ $V$S$P$S$G$E[m$E[m
set include=e:¥dvlp¥xc1¥include¥
set lib=e:¥dvlp¥xc1¥lib¥
set INCLUDE=e:¥dvlp¥xc1¥INCLUDE¥
set LIB=e:¥dvlp¥xc1¥lib¥
TMSIO
DCACHE C: E: F: H: -M2024
F:
CD¥USER
goto END

:MAXMEM                                    ----- MAXEMM.SYSで起動した場合
9SCDRV -R                                  ----- メモリを確保するため、切り
                                                  離している
F:
CD¥USER

:END
```

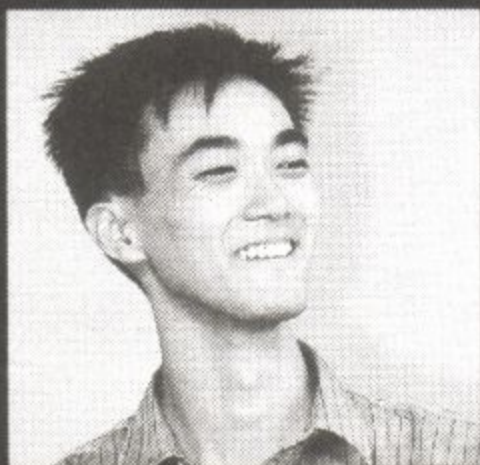
以上のような形でDINIT.SYSとINICHK.Xを組み合わせて使用することで幅の広い環境を選択することができるようになります。メモリをフル実装し、ふだんはメモリが余っていていつも快適な環境でX68000を楽しんでいる方にはあまり必要なプログラムではないかもしれません。しかし、少ないメモリを有効に使いつつ、できるだけ快適な環境を得たいと考えるのならば、このプログラムは必須のものではないでしょうか。

第3章

# ぼくらは、なぜフリーソフトを作っているのか

司会　郁

松戸市を中心とした草の根ネット、ひるまネットのSys.op.。
本書のスタッフの1人として第2章のフリーソフトの紹介も担当している。

**YuNK**
本名　中村祐一。1970年、神奈川県生まれ。
パソコン歴：小学校のとき、パソコンに触ったのがきっかけで、以来10年。
X68000歴が2年半。
パソコン通信歴：1年。
自作フリーソフト：HIOCS、HAS他。

**星野美季**
本名　望月　明。1964年、静岡県生まれ。
パソコン歴：はじめて使ったのがTK80、以来17年。X68000歴は初代機からのつきあいで、1年ほど前にXVIに買い換え、通算6年目。パソコン通信歴はMZ-2500で始めたので、7年くらい。
自作フリーソフト：Telecom Miki他。

**西表山猫**
本名　奥津　篤。1963年、神奈川県生まれ。
パソコン歴：はじめてのパソコンはカシオのポケコン、以来10年。
X68000歴は3年。
自作フリーソフト：TMN

**Ext**
本名　川本琢二。1962年、大阪府生まれ。
パソコン歴：9年。通信歴は4～5年。
X68000歴は5年目。当時88VAを買おうか、X68000を買おうか迷った末、MC68000が載っている日本製パソコンが欲しいということで決めた。
自作フリーソフト：DEDIT、TwentyOne他

日時：1992年10月17日　場所：ソフトバンク

## はじめはみんな初心者だった!?

**郁**　本日は、遠いところをわざわざおいでいただき、大変ありがとうございました。これからX68（X68000）用の代表的なフリーソフトを作っておられるみなさんに、フリーソフトの作者として、なぜフリーソフトを作りはじめたのか、製作の裏話、サポートの苦労話などを話していただければと思います。

今やパソコン通信の世界では有名なみなさんですが、最初からバリバリだったわけではないと思います。この本の読者も、これからパソコン通信を始めようという人たちなので、まずは、パソコン通信を始めた頃のみなさんの苦労話などをお聞かせいただくと、読者も安心できるのではないか（笑）と、思います。その頃のことを思い出しながら、話していただけますか。

**Ext**　私の場合は、身近に通信をやっている人がいて、「お前もやれ」って言われたから始めたんです(笑)。その頃、今のDEDITの原型ともいえるプログラムを、ある雑誌に投稿して掲載料が入って、ちょっとリッチになったものでモデムを買いまして。

**郁**　その頃のボーレートはどのくらいでしたか。

**Ext**　2400です。そのとき買ったモデム、そのまま今もずっと使っています。だから、300bpsとか、その頃のことは全然知りません。画面を見せてもらったりしたことはありましたが、そのときは通信には全然興味がなくて、あとで、「あぁ、これが通信というものだったんだ」という感じでした。

通信ソフトは、まわりにいたパソコン通信の先輩が全部揃えてくれました。やっぱり、勧めた手前ね(笑)。だから、別にパソコン通信をやりはじめて困ったということはなかったんです。

**郁**　じゃ、その先輩がいらっしゃらなかったら、通信はしていなかった？

**Ext**　そうですね。通信なんかしてなかったでしょうね。まあ、今はわりとポピュラーですから、自分で興味を持って、やっていたかもしれないけれど。

**郁**　その頃は通信をやるというのは珍しいことだったんですか？

**Ext**　私にとっては、もちろん、珍しかったですね。でも、わりと広まりつつあったんじゃないですか。

**西表山猫**（以下、山猫と略す）　どこのネットですか。

**Ext**　一番最初はNIFTY（NIFTY-Serve）です。XOS9（OS-9/X68000）のEXSCFというライン入力のドライバを作ったのを、通信をやっている別の人に代理アップしてもらったというのが一番最初ですね。

**郁**　最初からプログラムをアップするために、ネットに入ったというわけですか。

**Ext**　そうです。

**星野美季**（以下、美季と略す）　私の場合は、パソコン通信を始めた頃というのが、本当にパソコン通信がまだ創成期だった頃でして。その頃、ある会社でネットをやっていたんです。草の根ネットですが。MZ-2500が出るときに、シャープが大々的に販売店でパソコン通信のホストを始めようということで、ホストプログラムをまいたわけです。以後、それでやりはじめて、パソコン通信のホストをやっていたら、なんとなく通信っておもしろそうだなというわけでね。

だから、その当時は、ホストでアクセスばかりしていたんです。だけど、やっぱり自分でも電話を通してやってみたいなということで、パソコン通信を始めたんです。まだ、その当時は本当にママゴト程度で。X1turboか何かで、BASICで通信ソフトを作ってましたね。まあ、単純にネットのメッセージを読んだり、書いたりするのが楽しかった時期です。

その後、68を買って、68だと、その当時、通信ソフトが福袋にもついていなかったし、あの頃の市販のものといえば「X LINK」ですか。今はもうなくなってしまいましたが。その後少したって、「Communication PRO-68K」が出て。まあ、そういうソフトはあったんですが、まだ通信ソフトらしい通信ソフトがなかった頃で、ある雑誌に通信ソフトのソースがアセンブラで載ったんですね。福袋でアセンブラがついてきまして、それを使っていたんですが、この通信ソフトではテキストのアップ／ダウンしかできなくてね。バイナリが何も使えないわけです。いろいろ不便があるなということで、自分で通信ソフトを作りはじめたんです。

注●X LINK：市販されたX68000のはじめての通信ソフト。
●Communication PRO-68K：シャープ純正の初心者向け通

信ソフト。画面が派手。全画面がバックログに使用されるという驚異の仕様だった。

その頃は、まだネットワークってあまり頻繁にはやっていなかったんですけれど、まあ、300bpsですから電話代もすごくかかりましたね。その後、1200bps、2400bpsになってきた頃に、パソコン通信もおもしろそうだなということで、自分でそのとき作っていた通信ソフトをアップしはじめたのがパソコン通信でアクティブになったきっかけです。

戸惑ったことというのはいっぱいあって、ish（第2章122ページ参照）をダウンするのに、ishがかかっているものをダウンしてしまったとかね（笑）。

最初、8ビットマシンでやっていたときに、CP/M上でISH.COMというのがあったんですが、これをダウンするのにどうすればいいかわからなくてね。ダウンしたら、なぜかish.ishだったんです。それを解凍するのに、またishをダウンするはめになったという、ちょっと間抜けなことをやりまして（笑）。

パソコン通信だと基本的にいろいろな人と話せますし、NIFTYなんかだとRT（リアルタイム会議）なんかでExtさんたちとも話させてもらえますし、いろいろ勉強にもなりますね。

注●RT（リアルタイム会議）：NIFTY-Serveでのチャットの呼び名。チャット→342ページ参照。

あと、自分にとってみると、通信というプログラムの発表の場が身近にいっぱいあるわけです。私なんかは自己顕示欲が強いほうなんで、くだらないものでもなんでも、とりあえず作ったらネットにアップするというようなイメージで、今はやっています。だから、そういうところがすごく役に立っていますね。今は確かに通信ソフトもいっぱいありますし、まわりに通信をやっている人もいっぱいいますから、これからパソコン通信を始める人というのは、そういう意味では、かなり楽に入ってこられるんじゃないですか？

**山猫**　私の場合は、隣にいる美季さんに通信でもやってみないかと言われまして。使い古しというか、使い回しというか、シャープが出していた手動の300bpsのモデム、MA1X19-22という手動のモデムをもらいまして。その頃持っていたX1turboで通信を始めたんです。

300bpsで、たらたら画面に流れてくるのを見て楽しんでいる時代でした。何をやってもおもしろいという時期ですね。ちょうどパソコン通信をやる10年ほど前にアマチュア無線をやっていまして、はじめて声を出したときの感動というのが蘇りまして。「あっ、これはおもしろい！」ということでのめりこみました。一番多いときには電話代だけで9万円くらいかかりましたね（笑）。

はじめてパソコン通信をやったときは、以前アマチュア無線をやっていたころ、はじめて声を出したときの感動が蘇りまして……（山猫）

パソコン通信で戸惑ったことというのは、ほとんどなかったんですけれど。自分のマシンで扱いきれないほどの量のソフトをダウンしちゃって、解凍できずに困ったとかね。そういうことはいろいろあります。

**YuNK**　ぼくの場合は、通信を始めるよりは68を買ったほうが先なんです。68を買ってから、ちょこちょこプログラムを組んだりしていて、最初にハイスピードアセンブラ（HAS）を作って某ディスクマガジンなんかに発表しまして。採用されたらいいかなという程度に思っていたんです。

注●ハイスピードアセンブラ（HAS）：YuNKさん作のフリーソフト。AS.Xより高速で、バグが少ない。

それで、大学の友達に「こんなのできたんだけど……」みたいな感じで配っていたら、友達や先輩とかでけっこう通信やっている人が多くて、いつの間にか先輩や友達を通じて（HASが）ネットにアップされていたんです。で、先輩や友達から「こんな要望があったんだけど……」というのを手渡されて、そのレスのファイルをディスクに詰めて大学に持っていって手渡すということをしばらく繰り返していたんです。そうやって、バグの発見とか直しをしたりしていたんです。

注●レス：レスポンス(Response)の略。書き込みに対する返答。

だけど、それも使用者の要望とかが伝わってくるのに時間がかかるし、なんとなくもどかしいし。それに、前から通信もやってみたいなと思っていたのに加えて、ぼくの友達で通信を始めてズブズブにハマっていたのがいまして、そいつがぼくの下宿にモデムを持って押しかけてきて、「こんなに通信はいいものだぞ」みたいな感じでやって見せてくれたんです。それで、始めてみようかなということで始めた

のがきっかけですね。

ぼくに通信を強力に勧めた友達は、もう完全にハマりこんでしまって、自分でネットを開局するまでになってしまいましたね。彼は、今、八王子にあるネットのSys.op.をやっているんですが、最初は、その友人の開局したネットと、あとは梁山泊とツクモネットですか、そのあたりに出入りしていましたね。しばらくしてから他のネットに出入りするようになりました。とにかく68関係だとNIFTYが強力だという話も聞いていましたし、なんとか電話代もこのくらいだったら払えるんじゃないかなと思ったので、あとでNIFTYにも入ったという感じですね。

注●Sys.op.：BBSの管理者。草の根（無料）ネットでは主宰者が兼ねていることが多い。
●梁山泊：X68000のパワーユーザが多いネット。最近はAMIGA、MAC、IBMのユーザも増えている。
●ツクモネット：九十九電機が主宰するネット。X68000の話題が豊富である。

**郁**　ありがとうございます。では、みなさん、パソコン通信に参加して、友達が増えたとか、役に立ったということはありますか。

**Ext**　やっぱり、友達の数はすごく増えましたね。だけど、大抵みんな東京でしょう。だから、あんまり知らないけど、知ってるという感じですね(笑)。そのせいで、東京へ来やすくなって、なんか出張があると必ず１日延ばしたりして（笑)。

**郁**　Extさんは名古屋の方ですが、NIFTY以外の活躍の場が名古屋あたりにもあるんですか？

**Ext**　ええ、通信を始めたばかりの頃は地元で、草の根ネットでやっていました。でも、私の場合、通信を勧めてくれた人がNIFTYにIDを持っている人だったんで、とにかくNIFTYで通信を始めることにしたんです。だから、モデムを買ったら、すぐにNIFTYのIDをとりました。だけど、NIFTYの会員になるまでに時間がちょっとあるからというので、草の根のほうでしばらく遊んでて、その後、NIFTYにどっと流れこんだというわけです。

**郁**　草の根とNIFTYみたいな大きな商用ネットだと、かなり雰囲気やサービスも違うと思うんですが、そのへんはどうですか。たとえば、草の根の場合だと、地方が多いですよね。話題なども違いますか？

**Ext**　私はコンピュータ以外の話はしないから、わからないです。だけどまあ、地元のネタが集まりやすいということはありますね。

## ネットとのつきあい方

**郁**　初心者へのアドバイスなんてありますか。

**Ext**　アドバイスなんて、そんな大それたこと(笑)。アドバイスということではないですけれど……。うーん、やっぱり、ないです（笑)。

**郁**　美季さんは、どうですか？

**美季**　まあ、初心者へのアドバイスというわけではなく、ネットをやっている人全員に対してだと思うんですが、やっぱりネットって自分で書き込んだり、アップしたりするのがおもしろいんだと思うんです。もちろん、いっぱいダウンロードするのは、それはそれで問題ないと思うんですが。でもやっぱり、自分で書き込んだほうが楽しいと思いますね。どんなくだらないことでもいいから、とにかく、まず書いてみるということを実践するといいんじゃないかなと思います。

**山猫**　はじめてネットにアクセスされる場合、通信のホストというのは１つの社会を形成しているのだということを忘れなければ、大変楽しめると思います。つまり、ネット上には文字しかないから、本来の自分とは違った自分を見せることもできるんです。すると、中には羽目を外しすぎちゃう人もいて、結局、そういう人は通信をやめてっちゃうんです。でも、こういう楽しみ方だと不幸ですね。せっかく楽しくやるつもりならば、ネットが１つの社会であるということを自覚しておく必要があると思うんです。

**Ext**　今の話にあった「ネットは１つの社会である」というのと同じですが、結局、人間関係なんですよね。特に有料ネットの場合だと、お金を払って情報を得ているという感覚があるでしょう。だけど、そういう考え方をしていると、情報源も私たちユーザのボランティアであるんだというのが全然見えなくなっちゃうんですよ。つまり、NIFTY代は電話代と同じように情報伝達料というような感じで見て、あくまで情報はタダでもらっているんだということを頭に置いておかないとね。結局、人と人とのつきあいでいろいろ会話をし、情報を得ているんだという、そんな感覚を持っていればうまくいきますよ。

## 通信の世界での
## ギブアンドテイクとは？

**YuNK**　それと、通信でもフリーソフトの利用の場合でも、ギブアンドテイクというのが、やっぱり大事ですよね。

**Ext**　中にはギブしたくても、「プログラムではどうも……」って人もいますよね。そういう人はそういう人で他のギブのしかたを考えてもらえばいいわけで。

**YuNK**　プログラムの形じゃないギブもできるわけですからね。

**Ext**　そう、そう。今までずっともらってばかりで悪いなと思っていた人が急によく書き込みをするようになったんです。で、聞いてみたんです。すると、「もらってばかりで何かしないといけないなと思ったから、××のアップのリストを作ってみました」というんですね。そんな情報もうれしいですからね。ああいうのを見ると、いいなと思ったりしますね。

**郁**　そうですね。ギブも方法がいくらでもありますから。ただの感想だっていいわけですし。確か山猫さんは、TMNで絵はがき……なんていうんでしたっけ。

**山猫**　メイルウェアみたいな。

**郁**　絵はがきウェアみたいなことをやっていましたね。あれはどうですか。

注●絵はがきウェア：通信等で出回っているソフトの中には「使用してみて気に入ったら何か送ってください」という一言がついてくるものがある。それを通常「＊＊＊ウェア」(シェアウェアなど。シェアは作者と利用者で利益を分けあうニュアンス)と呼んでいる。TMNの場合は「絵はがき」を送ってください、と書いてあり、「絵はがきウェア」ということができる。

**山猫**　そうですね。これまでで全部で50通ぐらい来ているんです。意外に自分の知らないような草の根のネットまで転送されていて驚いているんですけれど。元のネタというのは、Macのソフトからなんです。よかったら、あなたの国の絵はがきを送ってください、という海外のフリーソフトがあって、これはいいやと思って真似してみたんですね。意外に反響がいいんです。いい絵はがきがないからといって、自分で撮った写真とかイラストとかで送ってくれる方もいます。その土地のおもしろい絵はがきをいただいて、ひじょうにフリーソフトを作っていてよかったな、という感じはありますね。

結局、人と人とのつきあいで情報を得ているんだという感覚を持っていればうまくいきますよ(Ext)

**郁**　やっぱり反響がないと作りがいがないということですか？

**山猫**　ええ。それにネットにプログラムを上げると、大抵の方がオンラインで書き込みされるので、思いつきでパッと感想を書かれて、たまにちょっときつい言い方をされちゃうことがあるんです。でも、絵はがきだと柔らかく、バグがあっても柔らかく、書いてくれるので（笑）。その点、楽です。

注●オンラインで書き込み：電話回線をつないだまま書き込みをすること。

## バグ情報としてのレスポンス

**郁**　反響がなければ、もうパソコン通信はやれないようなことになってきましたが、美季さん、いかがですか？

**美季**　私の場合は、先ほども言いましたが、要はMikiをアップしたのも、最初は選択の幅を広げたいということだったわけです。使っている側も、市販ソフトばかりでなく、フリーソフトもいっぱいあるわけですから、その中から自分にあうものを選べばいいわけです。たとえば、MuTermを使っている人もいますし、WTCを使っていながらMikiも使っている人もいますし、TMNを使っていた人がまたMikiに戻ってきてくれる場合だってある。当然、ABtermを使っている人というのもいるでしょうし、TSterm2だっているでしょう。

注●WTC：PC-9801、FMR、J-3100等の機種で広く使われている通信ソフトにWtermがあるが、これは、その操作性を中心に同様のものを目指したX68000用通信ソフト。
●ABterm：X68000の黎明期からPEKINネットを中心に活躍していた通信ソフト。公開されたソースをもとに多くの人の手で改良が積み重ねられていった。
●TSterm2：去石氏作の通信ソフト。強力なマクロ言語を備え、質実剛健な印象。愛用者は多い。フリーソフトの

通信ソフトを挙げるとすれば、Miki（含むTMN）、MuTermと、このTSterm2の3つは必ず挙がるだろう。

そういうふうに、どんどん分解していって、その中で、みんな似たり寄ったりのところもあれば、全然違うところもあるわけです。その中で選択の幅を広げたいというのがあるんです。だから、私はあまりレスポンスというのはそんなに気にしていないんです。ただ、レスポンスの中で引っかかる部分というのがありましてね。フリーソフトというか、ソフト全体において、どうしても1人でやっていると、バグって取りきれない部分がありますよね。ですから、バグ報告って、一番ありがたいものなんですね、レスポンスの中では。

ただ問題は、その言われ方がたまにククッと引っかかるものがあったりするんです。だから、すごく厳密に言うと、バグが出たときの状況から、そのとき組み込んでいるものすべての情報を、とにかく詳しい情報がもらえればベストなんです。最低限でも、「こういうふうな感じで、こういう操作を行っていったら、こうなっちゃいました」くらいの情報は欲しいですね。一応、「こういうものを組み込んでおいたんですが」くらいの報告があると、こっちとしても何が問題なのか探しやすいんですけれども。

前にどこかの草の根ネットでもらったレスポンスでみんなで大爆笑したのが、いきなり「文字が出ません」と来たんです（笑）。それしか書いていないんです。「文字が出ません」といっても、「じゃあ、私はどうやってアップロードしたんだろう？」と思うわけでして。そういうのが来ると、確かに困るというところもありますね。その反面で、逆に「使ってます、ありがとうございます」というようなことを言ってもらえると、やっぱりうれしいですし。まあ、レスポンスは、あればあったでうれしいんですけれどね。

ただMikiに限らず、私が作るものって、けっこうくだらないものが多くって。前に作ったのが画面のバックグラウンドで絵を描こうというものだったんです。ただ単にマウスのポジションで線を引くという、すごくくだらないものを作ったんですね。反響はほとんどなかったんです。ところが、たった1人だけ、コンパイルの間、暇じゃなくなったというのがありましたね（笑）。

まあ、そのプログラムはバックグラウンドに常駐して、絵をぼかぼか描けるんでね。コンパイルやっているときでもなんでも絵を描いちゃいますから。だから、暇つぶしにちょうどいいというレスが来て、「うーん、くだらないものでも、レスポンスはあるものだなあ」と思ったことがありますよ。

**Ext**　それってセーブできるの？

**美季**　全然できない。ただ絵を描くだけ（笑）。線を描くのと消すことしかできないんです。ただ、それだけなんですよ。そういうくだらないものにレスが来るとは作ったほうも全然予想もしていませんでしたから。ただ、「こういうものを作ったから、ハードディスクの肥（こや）しにでもしてくれ」と。「暇な人は組み込んでください」ということでアップしたんです。私は、そういう小さいものをよくアップしたりするんです。だから、そういう意味では、あまりレスポンスとかには期待していないんですね。

## 一番助かるレスポンス

**山猫**　私の場合は、レスポンスというと出会いとか、そういうのと関係するんです。一番助かるレスポンスというのがありまして。MAX BBSに行っててよかったところはその1つなんですが、たまにメイルが届くんです。メイルの中身がCのソースなんです。で、ここここにバグがありました、と。

**YuNK**　すごい！

**山猫**　1人で追いきれないところとか、わかっていても直せないところとかありますしね。あと、だいぶ深いところまで行きますと、プログラムの作り方の思想みたいなところで違う発見というのがけっこうありまして。それは1人だと絶対にできない部分なんですね。

で、フリーソフトの場合、プロの作るソフトとか、雑誌に発表するソフトなんかとは大きな違いがありますよね。みんなで作っているという点が。結局、コンパイルしてアップするのは1人の人間なんですけれど、それを環境として育てていくのは使っている人みんななんですね。そういう出会い方というのは、非常にうれしいですね。

**郁**　初心者へのアドバイスというのがあったら挙げてもらえますか。自分の場合、すごく苦労したけど、

このへんだけは気をつけたほうがいいとかね。やっぱり、通信する人が近くにいるというのがベストですか？

**YuNK**　それが、やっぱりベストじゃないですか。でも、こればっかりは環境の問題だから。そういう環境にない人はどうしようもないですけど。

**美季**　通信を始めるために、環境を整えるわけにもいきませんからね（笑）。

**Ext**　まあ、初心者が通信を始めるという意味では、もうちょっとネット側が親切になってもいいかもしれないね。NIFTYのFSHARPはちょっと不親切ですからね。さっき話があったけれど、ish.ishなんていうのも、けっこう困るところですよね。

注●FSHARP：NIFTY-Serveのフォーラムの１つ。シャープのパソコンに関心がある人たちが集まっている場所。現在、分野別に、FSHARP１、２、３の３つのフォーラムに分かれている。

**郁**　ネットの対応にも問題ありということですか。

**Ext**　ただNIFTYも、商用といっても、あれも実際にやっているのは草の根みたいなものですから。

**美季**　巨大な草の根っていうイメージですね（笑）。

**Ext**　集まっているということに対して10円払っているわけですよ。

**山猫**　そういうたとえで言えば、NIFTYの場合はフォーラムという形で全部分かれていて、ちょうど集合住宅みたいな感じですね。

注●フォーラム：NIFTY-Serveでは、あるテーマにそって独自の運営を行うサービスを「フォーラム」と呼んでいる。コンピュータ関連だけでなく、映画、スキーなど200種類近くのフォーラムがある。各フォーラムごとに会員が自由に書き込みができる掲示板、フリーソフトなどを収録したデータライブラリ、会員同士がリアルタイムにお喋りできるリアルタイム会議などで構成されている。

**Ext**　場所提供というような感じね。だから、あれも草の根みたいなものですから、あまり親切じゃないんですよ。というか、FSHARPの（広い会員層に対し）Sys.op.がそこまで手が回っていない。逆に、実際現場でやっている人に、もうちょっと親切にして欲しいなと思うわけで。

**美季**　不親切なところって、ちょっとありますよね。特に68の場合だと、市販の通信ソフトが、今、１つか２つしかないでしょ。だから、今、通信している人は、一体どうやって通信ソフトを入手したんだろうなということのほうが、私はすごく疑問なんですけれど。だから、今回、この本にいろいろソフトが載るそうですから、少しは通信人口が増えるんじゃないかなと期待しているんです。

「うーん、くだらないものでも、レスポンスはあるものだなあ」と思ったことがありますね（美季）

**山猫**　私宛のレスポンスの中で、「はじめて通信をしました」という人がいまして、私としては、いきなり、こんな難しいソフト（TMN）を使って大丈夫かなと思うことがあるんですけれど。要するに、友達にフロッピーで通信ソフトを渡されたというのがけっこういますよね。これが68じゃなくて、一番売れている98（PC-9801）の場合ですと、市販ソフトを買って移行していくみたいなのが多いようですね。68の場合は、やはり初心者の方は必ず誰か知り合いがいないと、パソコン通信なんて始めないんじゃないかと。そもそも68を買うこと自体、知り合いがいないと、たぶん、買わないんじゃないかと思いますし（笑）。まわりにそういう人がいれば、初心者が苦労するというのはあまりないんじゃないかなと思うんですが。

**美季**　確かにアクセスを始めてから、ネットになじむか、なじまないかというところで苦労することが多いと思いますよね。

**Ext**　そういう場合は、やっぱり、あちこち渡り歩くという……。

**美季**　１つや２つは、けっこう相性のあうところもあるもんですからね。

## 恐るべきは、電話代なり

**YuNK**　あとは、もっと根本的な問題なんですけれど。みなさん、お勤めされていますよね。ぼくはまだ学生なんですが、学生の立場から言いますと、電話料金には気をつけようというのがありますね。

**山猫**　それは、勤めていても同じです。

**YuNK**　ぼくはまだ下宿していて、アルバイトとかをやってどうにか暮らしているんですが、アルバイトをやっていれば通信の電話料金もどうにかできそうだということがあるんです。でも、ぼくのまわりには、まだ学生で親といっしょの電話機を使っていた

りするケースもたまにはあるんです。で、通信をやって長い時間電話を占領していると、親に白い目で見られたりとか、そういう話もあるみたいですしね。

**美季**　一時期、月に20万くらい使っていた人、いましたからね。

**Ext**　長電話でもちょっとないね。

**美季**　確かに電話料金には気をつけたほうがいいかもしれない。

**山猫**　余談になりますが、家で通信をやっている方はキャッチホンは外しておいたほうがいいですね。

注●キャッチホン：NTTのサービスの1つ。電話を使用中に他の人から電話がかかってくると電話がかかっていることを知らせてくれる機能。通信中にこの機能が働くと、ネットワークとつながっている回線が切れてしまうため、ネットワーカの天敵(?)と言われている。

**郁**　ちなみに、みなさん、最高にかかった電話料金はどのくらいか、お聞きしたいんですが。

**Ext**　私はあんまりかからなかったほうです。というか、絶対東京にはアクセスしないというわけで(笑)。

注●東京にはアクセスしない：NIFTY-Serveのような大規模なネットワークの場合、各地にアクセスポイントを設け、地方の人の電話代がかからないようになっている。草の根ネットの場合は、このようなアクセスポイントを持っているものはほとんどない。

**郁**　それは賢いです。では、月に1万ぐらいですか。

**Ext**　いや、それでも、いつも2、3万はかかりますね。

**美季**　私は、一番多いときでも3万は超えた記憶がないですね。今はスケジューリングして、平均すると5000円から8000円の間に抑えるようにしています。だから、遠いところは1週間に1回で、近いところは極力毎日という感じですね。でも、やっぱり1万円以下というところで抑えています。でも、最高はやっぱり3万ぐらいいったことがあります。

**山猫**　初期の頃ですけれど、課金をあわせると最高9万円を超えたときもあります。で、それ以降は研究しまして、一応抑えているんですが、平均すると2万はいっちゃいますね。どうしても68系のネットの中心が都内に多くて、情報を集めるとなると都内にアクセスすることが多くて……。

注●課金：有料ネットの使用料。NIFTY-Serveの場合は、通常、1分アクセスするごとに10円加算される。この課金とは別に電話代がかかる。

いろいろ策を講じて、VANにもいろいろ種類がありますよね、Tri-Pとか、TYMPASに入ってみてとか。あと、日本高速通信とか、第二電電系に入ってみたりして、ね。今はもうセーブしてます。今、一番お金がかかって苦しいのは、NIFTYのアクセスチャージ（笑）。

注●Tri-P、TYMPAS：VAN業者。全国にアクセスポイントを持ち、VAN事業者と契約したパソコン通信のホストとの間を結ぶ専用回線を提供する。利用者は、手近なアクセスポイントまでの電話代とVAN事業者の設定する通信料金を払うことになるが、遠距離の場合、NTTの電話線で直接ホストまで電話するのに比べれば安くなる。

●日本高速通信、第二電電：NTT以外の電話会社。遠距離に電話をかける場合は料金が若干低めに設定されている。

●アクセスチャージ：課金と同意。

**Ext**　9万の請求書を見たときって、どんな気分だった？

**山猫**　電話の名義は親になっていたんで、親から見せられて、何も返す言葉がなかった（笑）。ちなみに、その9万の請求が来たときに、X1turbo、中古で売りました。電話代のために。

**美季**　私にはできないな、きっと。

**郁**　9万もかかったというのは、何に一番かかったんですか。チャットですか？

注●チャット：オンラインでする筆談。電話が1人の人間と話すものだとすれば、チャットは同時に何人もの人間とキーボードからタイピングしながら会話できる。

**山猫**　そうですね、チャットとか。

**美季**　確かにチャットをやりはじめると高いですね。NIFTYで会話してると、絶対3、4万楽にいくからね。

**郁**　チャットやっているだけだと、9600bpsも関係ないですからね。

**美季**　いや、逆にNIFTYは不利でしょう。もしROAD 3に行ってチャットしたら、すごい贅沢だと思いますよ。

注●ROAD 3：NIFTY-Serveの高速回線。INS-Cおよび9600bpsの高速モデムが用意されている。ただし、課金が1分あたり25円であるため、割高である。

**YuNK**　まだちょっとNIFTYでRTは怖くてできないですね。電話料金が、ね。

**山猫**　そういう人っていますよね。

**美季**　でも時間を、たとえば10分なら10分と区切ってね。

**YuNK**　区切れれば、いいですけれど。

**山猫**　いや、NIFTYの場合は、RTは（楽しくて）10分、20分でやめられません。(最初から)やめたほうがいいですよ。
**YuNK**　ぼくの場合、電話料金は、最高でNIFTYの課金とあわせても2万ぐらいかな。あまり行っているネットがそれほど多くないということもあるんですが。今だと、だいたい平均で1万2000～1万3000円ぐらいだと思いますけれども。まだおとなしいほうかな。
**郁**　そうですね。ほとんど都内ですか？
**YuNK**　そうですね。

## 自作プログラムを語る①
## －HIOCS－

**郁**　どうもありがとうございました。では次に、今回この本に収録されるみなさんのプログラムについて、少し話していただけますか。
**YuNK**　最近ぼくが作っているソフトというのは、前のアセンブラもそうなんですけれど、スピードを上げるのばっかりで（笑）。
　だいたいぼくがフリーソフトを作る動機というのは、自分が不満に思ったから作るわけでして。これは、みなさん、だいたいそうだと思いますけれど。
　最初は、純正のIOCSのスピードも、まあ、あんなものかなと思っていたんです。だけど、フリーソフトでHSTという、やっぱり高速化の文字表示ドライバがありますよね。あれに差し替えてみて、IOCSとはまったく速度が違ったので、これは一体どうなっているんだろうと思いまして。で、ちょっと中を覗いてみたら、なかなかすごいことをやっているんですね。
注●HST：文字表示を高速化するプログラム。
　でも、HSTにすると、それはそれでいいんですが、たとえば、文字のフォントのベクタだけ乗っ取るようなプログラムの場合だとHSTがきかないんですよね。直接フォントアドレスを求めるとか、読んでいるという感じで。そういうところとか、他のプログラムと組み合わせたときに若干不満があったんです。それとIOCSを使うとグラフィックのほうも高速化されますが、その場合はHSTとIOCSの両方を組み込まなくてはならないというのがなんとなく無駄なような気がしたので。それなら、もともと速いIOCSがあればいいんじゃないかと思って。
注●文字のフォントのベクタだけ乗っ取るようなプログラム：フォントを変更できるようなプログラムのこと。HIOCSがその1つである。
　で、IOCSの中身を調べて、どうやったら文字表示を高速化できるかを考えたのが最初なんです。それから、他にもROMのバグが報告されてきたのを直せないかとか、いろいろ作業も増えていって今の形になったというわけで。今もまだ不完全な部分があって、発展途上という感じなんですけれど。

最近ぼくが作っているのはスピードを上げるのばっかりで（YuNK）

**郁**　今も手直ししているわけですか？
**YuNK**　そうなんです。マウスのほうがうまくいかないんです。で、調べてみたら、マウスカーソルの表示ルーチンがおもりのように重かった、というのに気がついて。これを直さないと納得がいかないなあというので手を出して、今、ハマっているわけです。
**郁**　TCとかはどうですか。
注●TC：Turbo Consoleの略。文字表示を高速化するドライバのはしりとなったフリーソフト。
**YuNK**　TCもしばらく使っていたんですけれど、最近のバージョンはどうかよくわからないんですが、IOCSとそれほどスピードは変わらないなという感じがして。どうなのか、よくわからない面もありますが。
**郁**　自分ではHIOCSが最速だと思ってます？
**YuNK**　いや、そういうわけにもいかないんで。他のプログラムとの相性とかを無視すると、本当はもっとスピードは上がるんですが。でも、ほとんど自分の能力の限りでできることはやりつくした感じがありますね。
**郁**　やはり、他のプログラムとの相性とかはありますか。
**YuNK**　けっこう、あるんですよね。それと、HIOCSを発表して何ヵ月もたって、文字表示関係で思いもよらないようなバグが報告されたりとかもあったんです。もともとIOCSというのは、いろんなソフトの根幹の部分ですよね。だから、いろいろ微妙な部分

が難しいなという感じはありますね。
**郁**　ありがとうございました。では、美季さん、MikiとTMSIOのからみから話をしてもらえますか。

## 自作プログラムを語る②
## －Telecom Miki、TMSIO－

**美季**　私が作ったのがTelecom Mikiでして、今は分家が２人ほどいまして。１つが多機能版のTMNと、もう１つが名前は知らないんですけど、MAX BBSのY.さんが作ったTelocom Miki Nextの高速版というのがあって。
**郁**　「をたくバージョン」って言うんでしたっけ(笑)。
(会場で傍聴していたY.さん、苦笑)
**美季**　まあ、そういうふうに分家を従えるようにまでなりまして。Mikiを作りはじめたのが、そもそもさっきも言ったとおりで、通信ソフトがこれといってなかった頃でして。Mikiの一番最初のバージョンを見たことがある人って、たぶん、今、ここにはあまりいないと思うんですけれど。なぜ作ったかというと、その当時の68用の通信ソフトは色が出なかったんですね。デフォルトの４色、要するに黒と水色と黄色と白しか出ないとか。もっとひどいのになると、白黒しかなかった――シャープ純正ソフトですね――という状況がありまして。

で、ふっと隣を見るわけですね。98が狭い画面ながらもチカチカ点滅はするし、色は７色カラフルに出てるし。特に、その頃多かったホストのソフトがWWIVで、やっぱり、カラフルな画面なんですね。で、すごく頭にきちゃったわけです。なんで68000を積んでいる機械がこれだけしかできないのかって。そのときはあまりハードのことは知らなくて、某社の青本を読んでいたら、いろいろおもしろいことができそうだなということがわかって。

注●WWIV：パソコン通信のホスト用プログラムの１つ。

それが出だしでした。とにかく８色とリバースと、あと、ブリンク（点滅）が98並みにできるものを作ろうということから、まず出発して。次に、その当時、最初に使っていたソフトはバックログすら見れなかったので、やっぱりバックログも見られるようにしようと考えて、そこから始まったんです。その後、いろんなユーザの方に使ってもらって、レスが来まして。

注●バックログ：通信ソフトの機能の１つ。画面から上に消えてしまった文字を前に戻って読むことができる機能。

まず、速度の面でいろいろレスが来まして。遅い、とかね。その当時、Mikiは文字表示に関してはすごくズッこけたことをやっていまして、IOCSの、確かTEXTPUTか何かを使って、要するに文字をイメージで書くというようなノリでやっていたんです。でも、やっぱり、それでは速度が上がらないということで、文字出力から何から全部自作で作り上げていきまして。最近はやりのウィンドウも少し入れてみようかなということになって、今の形のようなものができてきたわけです。で、ウィンドウでバックログをやったりしてね。

最初はシリアル通信ドライバもメーカ純正のものでした。その後、ちょっと高機能なものが欲しくなったので、去石さんからAUXSYS2.Xをバンドル許可していただいて。でも、やっぱり他の方が作ったものだと、自分でこういうことをやりたいなと思っても、どうしても身動きがとれなくなるときがあるんです。で、シリアル通信ドライバのTMSIOができあがったわけです。

注●去石さん：ENV.R、TSterm2の作者。本書には、去石氏作のDRVが収録されている。
●AUXSYS2.X：去石氏作のシリアル通信ドライバ。TSterm2が使用している。
●TMSIO：美季さん作のシリアル通信ドライバ。Telecom Mikiだけでなく、TMN、QuTERMが使用している。

TMSIOは、最近はQuTERMなんかにもバンドルしていただいて、何か一人歩きを始めている部分がありましてね。とにかく、できるだけ小さく、あと、バッファの容量をなるべく簡単に変えられるように、というのを目標に作っていたんです。まだ速度の面では不満なところが多いんですが、ただ私としては、そういうスピードアップのほうはあまり得意ではないので、スピードアップはとりあえずY.さんとか、他の方に任せっきりにしちゃおうかなと。とりあえず問題提起のほうだけやってね。

注●QuTERM：SX-WINDOW用の通信ソフト。9600bpsで通信しても画面表示が追いつくという優れもの。

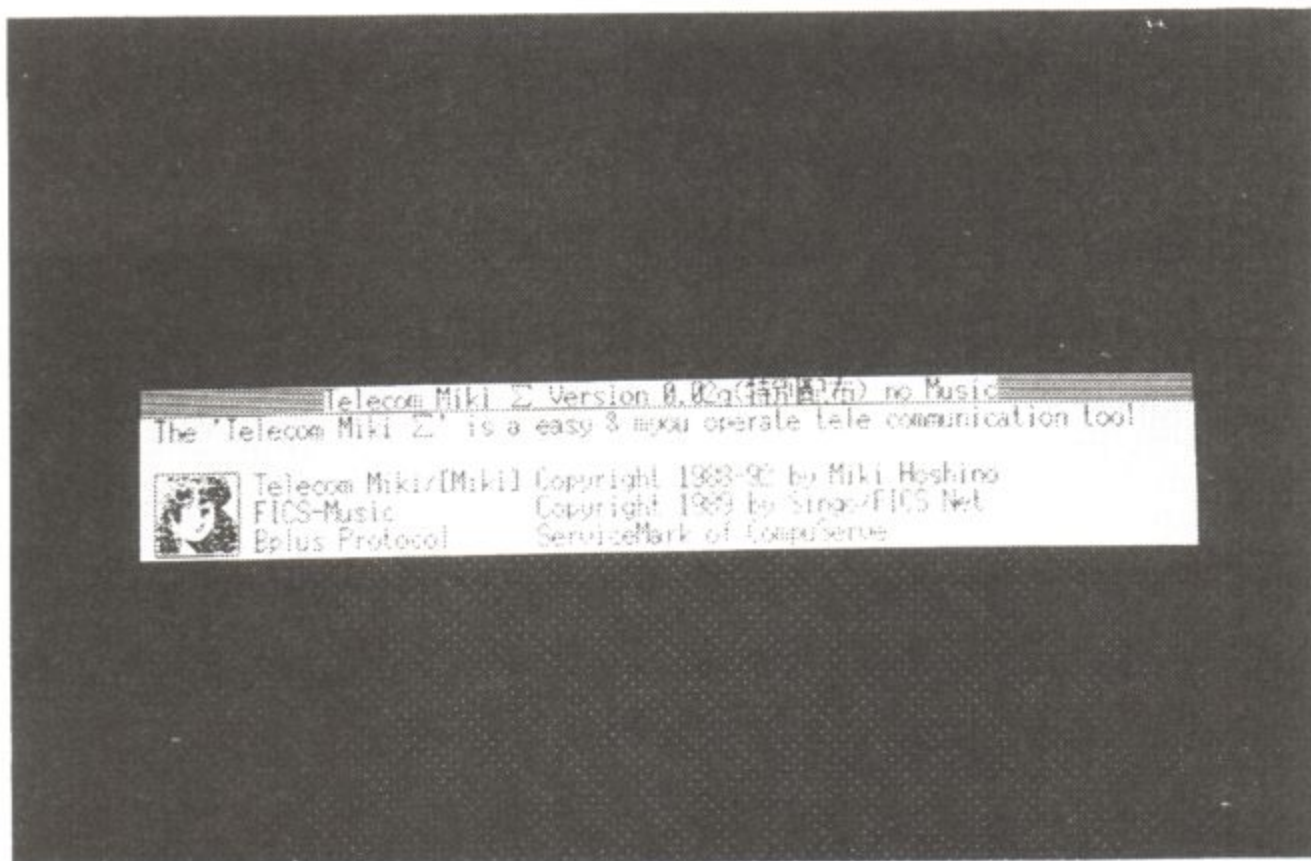
だから、Mikiのほうで自慢できるのは、X68000でバックでのスムーズスクロールを実現したことだと自分では信じているんですね。ああいう、すごく滑らかな、不気味だとも言われているスムーズスクロ

ールを実現させたわけです。

その後、VWがviewに名前が変わったときかな、スムーズスクロール対応になったんですが、あれを見ていると、画面の上からだいたい3cmから5cmのところにノイズが乗っているんです。これは、X68000でバックスクロールをラスターコピーを使ってやると、絶対出るんですけど、Mikiはそれを出さないというのが自慢なんです。

注●VW、view：ビューワソフト。X68000は、テキスト画面もビットマップ表示しており、文字表示が遅いと言われていたが、その常識を覆す超高速表示をしていた。

だから、画面がきれいだとか、そういう、訳のわからない、変なところで本家側がオタッキーに走っているんです。Mikiの最新版は、ある草の根で公開しているだけなので、見ていない方が非常に多いと思うんですけれど。最新版は、通信の、現在の各信号線の状況とかが画面で見られたりとか、本当にオタッキーなほうに走ってます。それに、画面表示がちょっと特殊で、今は34行画面で表示しています。

Pht.1　Mikiのタイトル画面

**郁**　どうしてですか。

**美季**　68のCRTCの表示行数はモニタが許す限りはいくらでも増やせますから。だから、今はデフォルトの32ではなく34行で、通信画面に31行使いながら、その他にステータス行や何かが3行も余分にあると。だから、漢字変換なんかも、それだけの余裕をもってできるわけです。だから、私が作ったソフトというのは、98に負けたくないというのがほとんどじゃないですかね。

私が作ったソフトというのは、98に負けたくないというのがほとんどじゃないですか（美季）

## 自作プログラムを語る③ ーTMNー

**郁**　それでは、続いてTelecom Mikiとのからみというか、どういうふうにつながってTMNになったのかというところを、山猫さん、お願いします。

**山猫**　まず、TMNという名前にしたのは、本家が復活するというので、同じ名前で出しているとまずいということで改名したわけでして。

**郁**　当初はどういう名前だったんですか？

**山猫**　当初は「Telecom Miki ε」という名前にして出したんですね。なぜ、同じ名前で出していたかというと、TMNを作りはじめたのは1991年の1月頃なんです。美季さんが、ある他のプロジェクトでプログラムを作ることになったので、もう当分は通信ソフトは作らないよ、と投げちゃったんですね。

「これはしまった」と。いろいろやってもらいたいことがあるんだけれど、今までは使う側ですから、言いたいことを、しかも、近くに（同じ会社ですので）いますから、「これが欲しい、あれが欲しい」と言いたい放題言っていまして。ところが、急にやらないと言われて、これは困ったと。で、美季さんに「ソースがあるから直してみなよ」と言われまして。ちょこちょこいじくりはじめまして。

ちょうどその頃、Mikiには確かXMODEMしか入っていなかったと思うんですね。あと、全部外部プログラムだったんですが、どうせやるのだったら、全部取り込んじゃって、かっこいいウィンドウを使いたいなと。そういう要望から、作ってもらえないものは自分で作っちゃえ。で、ソースを公開してもらっていて、しかも、改造もOKだよということもあったので、じゃあ、作ってみようという形で始めたのがきっかけですね。作りはじめたら、なぜかウケてしまって。そうしたら、もう抜け出せなくなっちゃって、現在に至っている形です。

注●XMODEM：バイナリファイルを転送するための通信規則。
●外部プログラム：Mikiは、XMODEM以外のバイナリ転送

のプロトコルを自前で持たずにプロトコルごとに外部の専用プログラムを呼び出していた。

本家のほうは高性能のほうに走ってるという感じなんですけれど、TMNはユーザの要望を取り入れて、なんでもかんでも、てんこ盛りにしちゃえということになってしまったわけです。その挙句の果てに、今、実行ファイルのサイズが220Kバイトを超えてしまったわけです。ちょっと通信ソフトのサイズじゃないな、というところまでいっちゃったんですが。

**郁**　TMNの多機能は山猫さんの趣味ですか。自分のわがままを通したとか。

**山猫**　作っている本人が使ったことがない機能というのがけっこうありますね（笑）。

**美季**　やっぱり、人からのメイルソースだよね。

注●メイルソース：メイルで送られてきたプログラムのソースのこと。美季さんの造語であると思われる。

**郁**　たとえば、どんなところですか。

**山猫**　メモ帳というのを作ってみたんですけれど、あれって、やっぱり使ったことがないですね。あるということをときどき忘れてますね。それくらいなので、大抵、自分が使わない機能というのはバグが多い、と（笑）。

**郁**　それはそうですね。では、もうそろそろTMNも終わりですか。

**山猫**　まだやりたいことって、いっぱいあるんですよね。当分抜け出せそうにもないし、まだ他にも作りたいものがけっこうあるし、どうでしょう。

**郁**　ありがとうございました。では、Extさん、TeXのほうもからめて、お願いします。

注●TeX：クヌース氏の作った組版システム。本来、組版は印刷会社の作業であるが、数学者でもある同氏が、自分の本を出版するにあたり、自分の思っているとおりに組版されないのに業を煮やして作り上げたと言われている。論文や学術書などがきわめて美しく仕上がる。ちなみに、『X68000 develop.』（ソフトバンク刊）はTeXで作成されている。「テフ」または「テック」と読む。

## 自作プログラムを語る④ -TwentyOne、DEDIT-

**Ext**　まあ、私は、移植だの、誰かの作ったやつを改造するというのはあまり好きではなくて、とにかく、今までにないもので何かおもしろいものはないかな、ということをいつもいつも考えているわけです。

たとえば、TeXのドライバというのは98にもありますが、私が作ったものとは全然違うものだと自負しています。また、Human68kはDOSファイルシステムとコンパチビリティを考えて作られていましたが、結局、あれはあまりコンパチビリティはないですよね。フロッピーを98に持っていって、小文字だったら駄目だとか、ハイフンは使えないとか。で、どうせコンパチビリティがないんだったら、もう徹底してなくしてしまえ、ということでできたのがTwentyOneです。

で、X68000の持てる機能を全部使えるようにしていくと、けっこうUNIXライクになったんです。すると、パワーユーザの方々がマルチピリオドを便利に使ってくれて。まあ、それならいいかなという感じですね。

注●マルチピリオド：ファイルネーム中に「.」が複数存在すること。例）TMN_ver.3.xといったファイル名が使用できるようになる。

**郁**　TwentyOneはどういった経緯で作ることになったんですか？　自分からとか、思いついたとか。

**Ext**　最初は、まがりなりにもファイル名に18プラス3文字使えるんだったら、きちんと認識して欲しいなということでね。だから、名前からして21文字全部使えるようにしましょうよ、ということだったんです。

**郁**　それはいつ頃ですか。

**Ext**　そう言われると……。他の人のほうがよく知っているんじゃないの？（笑）

**美季**　1988年じゃないですか。

**Ext**　88だ。

**美季**　88か89だと思うんだけれど。4年くらいたっているでしょう。私はデビュー当時から使っているから。

注●あとで確認したところ、1991年でした。

**Ext**　そうでしたっけ。だけど、あれは、ときどき困る人もいるみたいで、CONFIG.SYSに登録すると、動かなくなっちゃうとか、問題があるみたいで。まあ、いろいろ問題はありましたけれど、あくまでオタク対象ということでね。

**郁**　使っている人はみなオタクだと。

**Ext**　いや、だけど、オプションのつけ方によっては

普通の人が使ったりするんですよね。キャラクタならなんでも使えるよというオプションを使って、ネコの顔を描いたりする人もいたりしてね。

**郁**　ネコのキャラクタをファイル名に使っちゃうんですか？

注●ネコのキャラクタ：「(=^_^=)」とか「(@^_^@)」というように記号を組み合わせてキャラクタを表現する。

**Ext**　そう、ファイル名に使っちゃうの。自分のキャラクタとか言ってね。あ、もちろん、女性ですよ。だから、けっこう素人受けもするのかな、という感じも受けました。あれは、よかったですね。

**郁**　そういう使い方は知りませんでしたね（笑)。TeXのほうはどうですか？

**Ext**　TeXはやっぱり、98にもワークステーションにもない、いいやつを作ろうと思って作ったんです。

**郁**　68版だけですよね。Fontmanがあるのは。

注●Fontman：X68000のTeXでのFONTマネージャ。

**Ext**　はい、そうですね。

**郁**　それだけは誇れますね。

**Ext**　移植してくれという要望もあったりするんですが、もう、自分でソースを見てもわからなくなっていますから。

**郁**　みなさん、これだけ長くやっていると、自分のソースがわからないなんてこと、ありますか。

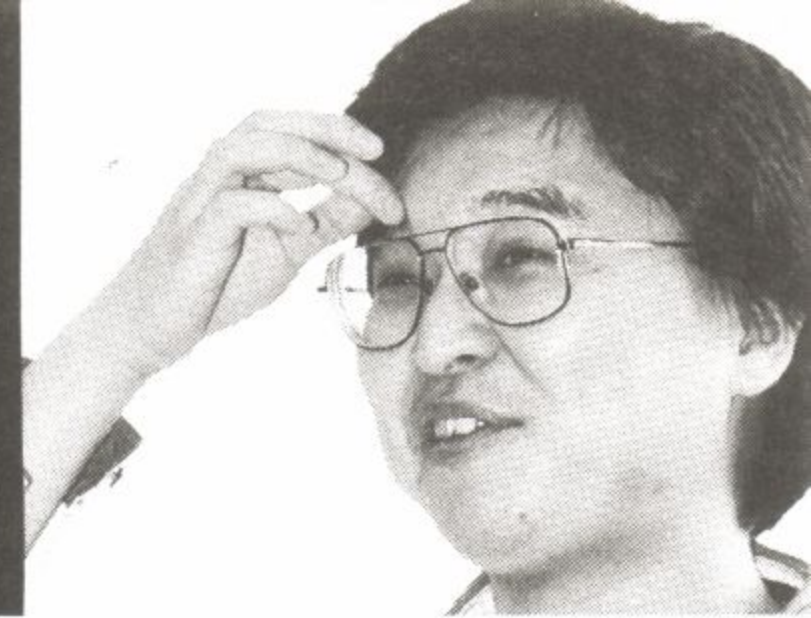
**美季**　Mikiではありますね。ソースの総行数がかなりデカいんですよ。無駄が多いんですけれどね。(山猫さんに向かって）Mikiを渡すとき、何行あったんだっけ。

**山猫**　あのとき、すでに35,000行くらいですね。

**美季**　30Kステップくらいか？　それくらいあるので、ときおり、なんでこんな変数宣言してるんだろうなって、ちょっと悩むときがありますね。でも、やっぱり30分くらいじーっと眺めているとハッと浮かんできますけどね。きっと私の頭、アクセスビリティが悪いんでしょう。

**Ext**　そういうことじゃあないけれど、このアルゴリズム、なんで必要なのかなとか(笑)、考えこんでしまうことってありますよね。もちろん、よく考えてみると意味があるんだけれど、そういうときに改造すると、絶対エンバグする（笑)。

注●エンバグ：ある機能にあったバグを取った際に、他の機能に新しいバグが入ってしまうこと。

（自分のソースを見て）このアルゴリズム、なんで必要なのかなと（笑)。そういうときに改造すると、絶対エンバグする（Ext）

## あなたはアセンブラ派？ それともC派？

**郁**　みなさん、開発言語は何を使ってやっていますか？

**Ext**　私の場合は、Cが90%で、あとはアセンブラですね。

**美季**　そうですね、本数でいったらアセンブラですね。常駐ものが多いので、小さく作ろうとするのがほとんどでしたから。ただ、全部のトータルKバイト数の割合でいっちゃうと8割がGCCですね。Mikiが一番でかくてね。やっぱり、GCC、HAS、HLKでやってますね。

注●GCC（GNU C Compiler）：GNUプロジェクト（第1章参照）の作品の1つである、Cコンパイラ。驚異的なことに、GCC自体マシン非依存の設計であり、各CPUのコード生成に関連したMD（マシンディスクリプションファイル）を与えることで、それぞれのCPU用のGCCができあがる。強力なオプティマイズ機能を持っており、X68000用に生成されるコードの質は、人間がアセンブラでゴリゴリ書いたコードと大差がないといわれている。シャープ純正のXCでは（Ver.1、Ver.2とも）勝負にならない。X68000用に移植されたGCCは、パソコン通信で配布されている他、『X68000 develop』にも収録されている。

●HLK：Hi-speed Linker。SALTさん作のLK.X上位コンパチブルなリンカ。LK.Xより高速でバグが少ない。

**山猫**　ほとんど99%がCですね。GCCがなかったら生きていけない。

**YuNK**　ぼくは、95%がアセンブラで、5%がCという感じだと思います。今まで発表したプログラムではCで書いたのって、たぶん、ないと思うんです。なんていうか、やっぱり、68000のアセンブラそのものにけっこう愛着があって。結局、自分で（HASを）作っているくらいですから。

**郁**　アセンブラがいいですか、やっぱり。

**YuNK**　なんとなく愛着があるなあと。でも、ハイスピードアセンブラ（HAS）のほうも、これから先を考えると、このままアセンブラのままで書くのは辛いかなという気はしていますけど。

**Ext**　アセンブラがソースを見てもわからなくなる言語の一番の候補じゃない？
**YuNK**　そうですよ、本当に。
**Ext**　Cだと、まだなんとかわかる感じだものね。
**YuNK**　HIOCSのほうはサブルーチンの集合体みたいな感じだから、1つ1つのサブルーチンはそれほど長くないので、まだあとで見直してもなんとかなるんですけれど。HASのほうは、あれ全体で1つのプログラムですし。今、ソースを全部あわせると250Kバイトぐらいあるので。Cで250Kならまだいいかもしれないけれど、アセンブラで250Kってすごいですよ。まあ一応、コメントは山のように入れてはあるんですけれど、それでもやっぱり、時間がたっちゃうとわからないですね。だから、バグ取りのときも、なんでバグが出るんだろうということよりも、ここは何をやっているんだろうということのほうに時間を費やしてしまって（笑）。
**山猫**　アセンブラ派とC派に分かれるところがあると思うんですけれど、68000のアセンブラって非常に作りやすいところがあるんです。作ってみると、確かに誰でもわかる。気にしなければいけないのはデータのサイズくらいですね。それさえ気にしていれば、あとは、これが欲しいというインストラクションがありますから、誰でも組めるんですけれど。やはり、アセンブラはある程度体系立てて作っていかないと、絶対ものができあがらないというところがありましてね。フリーソフトの場合、大体が思いつきで、その場限りのプログラムの気分で作ってますからね（笑）。

特にTMNなんか、実際に開発できる時間なんて限られていますから。本当にコードを書くのに10分とか20分でできあがらないとやっていけないところがあるんです。思いつきですぐに書けて、すぐに実行ファイルができて、確認できるみたいな形でないと、メンテナンスができないと思うんですよね。そうすると、どうしてもCのほうが記述性が高い。思いつきの記述性が高い。そのために、私はCのほうを多用するような形なんですけれど。やはり徹底的にやるんだったら、アセンブラのほうが楽しいですね。
**郁**　美季さんはどっちがいいですか。
**美季**　私は、頻繁に使っているのはアセンブラですね。まあ、68というのはハードがほとんどオープンで、どんなチップでも、大体、骨までしゃぶるくらい使えるので。ところが、そういうことをやろうとすると、やっぱりCだとどうしても追いつけないのと、あとはCだと、どうしてもスタートアップを組み込んだり、ライブラリを組み込んだりすると、どうしてもでかくなってしまいますからね。

特に常駐ものの場合、68はメモリが空いているからいいんじゃないかといえば、それまでなんですけれど。

たとえば、私、一定時間キー入力がなかったら画面が暗くなる、要するにスクリーンセーバで、NCというのを作ったんです。他のスクリーンセーバは大体5、6Kあるのかな。でも、私の作ったやつは、常駐部だけならば1Kを切りますから。だから、そういうコンパクトなのを作るときは、やっぱり小回りを利かせたほうがいいですしね。あと、チップを直接叩く（制御する）というのがすごく好きなんです。

おかげでMacintoshはいいものだということはわかっていても買わないんです。あれは、ハードを直接触ると怒られるでしょうから。そういうことで言えば、68はそんなことないですからね。だから、もうキーボードだろうとなんだろうと、割り込みから何から全部おいしくしゃぶりたいとなると、やっぱりアセンブラの右に出るものはないでしょうね。

C、これはもうCと言っていいのかな。GCCと言うべきですよね。私の場合、GCC専用命令ばっかり使っていますから。そういうことを考えてしまうと、他のCのコンパイルなんてできないですしね。まあ、どうしても、Cと言えばGCC主体になりつつありますよね。
**Ext**　アセンブラもいいですけれどね（笑）。Fontmanは常駐ものですが、あれは、実際自分自身をメモリの最高位……というような専門的な話をしていいんでしたっけ(笑)。最高位に再ロードしているんですよ。

だけど、あれ、Cで書いているんですよ。実際には、読み込みして再ロードしていくところはちょっとアセンブラを使っていますけれどね。本当にアセンブラで書かなくてはいけないところというのは、たとえば、高速性が必要なところにしても、私には結局、10％程度しかないんです。DEDITだったら、実際に画面にキャラクタを表示する部分だけでいいし、

その他のところはディスクアクセスを高速化したって、どうせディスクは遅いから。まあ、それなりのアルゴリズムは使ってありますけれど。本当に必要なところだけそうして、あとはCとアセンブラがおたがいに呼びあっていると、そんな感じでやっていて、90%はCというわけです。

あと、私は別に言語は選びません。アセンブラもおもしろいし、一方で、C++ や BISON、FLEXといった高級言語は、それはそれでまた、別のおもしろさがあるでしょう。両方、楽しいなという感じで。

注●C++：オブジェクト指向のC言語。
●BISON、FLEX：GNU版のYacc、Lex。構文解析、字句解析のプログラムを生成するためのツール。

**美季**　やっぱりGNUになるんですね。

注●GNU：「ジーエヌユー」または「グニュー」と読む。FSF(フリーソフトウェアファンデーション)のプロジェクト名。詳しくは第1章15ページ参照。

**Ext**　それはUNIXの世界でもそうだけれど、X68000の世界ではGNU以外はツールじゃないと(笑)。

**美季**　そこまで期待しちゃう？

**Ext**　言い過ぎですね。フリーソフトはそうですよ、いいですよ（笑)。フリーソフトの頂点にGNUはある。

**YuNK**　当然、GNUだってフリーソフトであるわけですからね。

## 最短プログラミング時間を競う

**郁**　先ほど、山猫さんは10分くらいでプログラムを作るとおっしゃっていましたが、他の人もそのくらいでプログラムを作ってしまうんですか？

**Ext**　10分ていうと。

**山猫**　思いついて。

**Ext**　思いつきの10分。

**山猫**　10分で思いついて10分で完了するわけじゃなくて、思いつき10分、コード決定に10分。1時間はすぐかかってしまいますね。

**美季**　私なんか、常駐ものでNCを作ったときなんかは、その頃、もう常駐もののスケルトンができていたので、中身のインタラプトのハンドラの部分を書くだけだったので、30分くらいでしたね。それも、ネットでチャットをやっているときにね。ハッと暗くなるのが欲しいと思って、一気にコーディングして、30分後にはネットにアップしてましたから(笑)。アップしたとき、まだそこでチャットを続けてましたね。で、「今、アップしておいたよ」と言ったら、「何を？」って言うから、「さっき言っていたやつ」(笑)。

注●スケルトン：似たような動作をするプログラムを作る場合、メインの処理の流れはだいたい同じになる。このような共通の部分を骨格とし、異なる処理の部分を肉づけするというプログラミング手法があり、この骨格部分のソースを「スケルトン」と呼んだりする。

フリーソフトの場合、大体が思いつきでその場限りの気分で作ってますからね（山猫）

常駐ものだからスケルトンがあったので一気にできましたけれど、でも、やっぱり1から作るのだったら、最低でも1時間ぐらいはかかるんじゃないですかね。思いつきでも。

**Ext**　それならCでも言えるよ。ディスク関係でちょっとトラブルがあったって言うから、RTに連れ出しておいて、「じゃあ、ちょっと読み書きしてセーブするっていうのを作ってあげよう」と言ってね。10分か20分でしたね。もうパターン決まっているしね。けど、Cですよ。Cでもそんなに早くできる。

**山猫**　私の場合はRTじゃないですけれど、NIFTYにバージョンアップしてアップロードしますね。あそこの場合にはフォーラムのSys.op.が確認して、OKなものだけ登録するという方式をとっていますよね。その反面、自分で削除できないということもあるわけで。で、実はアップを始めたら、その途中で改造を思いついたり、アップの最後のほうでバグが出ちゃったと。そういう場合、もう一度戻ってまた直して、アーカイブし直して再アップしたわけです。その間20分。同じバージョンのものが2本、実は陰に隠れていると。

**美季**　一番困るパターンだよ。

**山猫**　Sys.op.困らしてしまってね。

**美季**　メイル送った？

**山猫**　必ずメイル送ります。で、何時何分何秒のにしてくださいって（笑)。

**美季**　RTで言う場合もあるでしょう。「ごめんねー」なんて。

**YuNK**　NIFTYの場合はそういう感じってありますよね。だから、ぼくなんか、アップするのもけっこう慎重になってしまって。これはバグが出そうな気がするなと思ったら、NIFTYは会員数も多いですし、アップしたあとの影響が大きいですよね。だから、NIFTYはちょっとおいておいて、とりあえず近所の草の根にちょこちょこっと上げておいて、しばらく様子を見てから、このくらいで大丈夫かなと思ったらNIFTYに上げるという感じが多いんです。

**美季**　ロケテストはやっぱり必要だね。

**Ext**　ジャンクは？

**YuNK**　ああ、そうですね。ジャンクという手もあるんですよね。

注●ジャンク：FSHARP1にある「ジャンクショップ」という会議室のこと。通常の会議室と同じなので、プログラムはishでテキストにして書き込む。メンテナンス（主に削除）をSys.op.の手を煩わさずに自分でできる反面、300行という制限があるので、大きなプログラムは多数に分割する必要がある。

**Ext**　ジャンクなら自分で削除できるでしょう。

**美季**　まあ、Mikiあたりになると、意外とでかいからね。

**Ext**　そう、でかいと大変だね。

## バージョンアップの流し方

**山猫**　私の場合、いっぺんにアップしないと、バージョンの同期がとれないときがあって。たとえば以前、自分でもネットをやっていたんですが、そのネットでデータライブラリを設けておいたんです。ところが、NIFTYに送っておいて自分のところにアップするのを忘れてしまって。NIFTYに上がっていたから、近所の人が当然うちのネットにもあるものだと思ってアクセスしたら、なかった。「なんでNIFTYにあるのに、作者がやっているネットにないの」ってことになっちゃって（笑）。

わりとバージョンアップが日刊化していたこともあったんですけど。まあ、俗称日刊バージョンアップというのはよくあるんですけれど。いろいろバグがどうしても抜けなくて、直した差分を1日単位でアップしていたんです。ネットごとに違うバグつけちゃうと、あるネットではバージョンが先に進んじゃうということになり、使う人を混乱させちゃいますので。そういう問題があるので、アップするときは一斉にする。バグがあったらあったで直していくと。

**Ext**　なんかサイクルでアップしているという人が多いよ。

**YuNK**　そう、そう。

**美季**　うん、今どうなってるか、新しいのと比較してみないとわからない（笑）。

**YuNK**　ぼく、それでちょっと前に失敗しちゃったことがあって、やっぱりHIOCSですが。最近のバージョンアップで、やっぱりまだ不安な部分があるので、NIFTYのほうにも最新のバージョンが上がっていないんですけれど。

注●これは座談会の時点でのこと。その後、NIFTY-Serveにも最新版はアップされた。

で、NIFTYにアップを遠慮していたのとあわせて、あまり頻繁に行っていないネットのほうはアップを控えておいて、毎日とか2日に1回くらいの割合で行っているネットのほうだけバージョンを上げたのをどんどんアップしていったんです。久し振りにたまにしか行かないネットのほうに行ったら、作者がそのネットにいるにもかかわらず、他の人が親切に転載してくれていて。それに対してバグ報告がじゃんじゃんついていて、「このネットに作者がいるんじゃなかったでしたっけ」とかいう話が出て（笑）。

あれは、ちょっとまいった。「どうもすみません。私が作者です」と言って、あわてて返事を書きましたけれどね。

**郁**　そういう経験ってありますか、みなさん。

**山猫**　そうですね。全部自分の行っているネットだから、そこに送らなきゃいけないというふうにしちゃうと、電話代をとてもじゃないけど賄いきれない。今はアップする拠点というのを設けて、何か組織的なんですけれど、「誰々さん、どこにアップしてね」という感じでお願いして。

梁山泊もIDいただいたんですけれど。最初は自分でアクセスしていたんです。だけど、都内だけで4ヵ所、5ヵ所回っていたら、それこそ大変ですから

ね。しかも自分もいけないんですけれど、TMNはでかいサイズですから。2400bpsだとフルキットをアップするのに1時間以上かかるとかね。それをやっていたらとても賄いきれないので、「転送お願いします」ということにして、そこから（梁山泊とかに）枝分かれするという感じで。

**郁**　山猫さんの場合、そういうところは4ヵ所ぐらいですか？

**山猫**　一応有料ネットが3ヵ所……(笑)。草の根がMAX、サンデーですね。

注●サンデー：テレビ東京系で放映されていたシャープ提供の「パソコンサンデー」というテレビ番組が母体となって、制作元のテレコム・ジャパン（現在はテレコムスタッフ）が運営しているネット。17回線ある。

## よく行くネットとアクセス回数

**郁**　みなさん、よく行ってらっしゃるネットを教えてもらえますか。いくつぐらいのネットに行っているかとかもあわせて教えてください。

**YuNK**　ぼくは、有料ネットではNIFTYだけですけれど。NIFTYは、そんなにいつも行っているというわけじゃないのですが。週に2、3回ぐらいですか。で、あとは特に頻繁に行っているのは、ぼくをパソコン通信に引きずり込んだ友人の開いた八王子の「みるきぃ・うぇい」というネットなんですが、あそこはほぼ毎日、あとは梁山泊も毎日か2日に1回くらい。その2つが特によく行っていて、他のネットは週1ぐらいという感じですね。

注●みるきぃ・うぇい：八王子を中心にした、あっしゅさんが主宰するネット。CG、アニメ系に強いネット。

**美季**　私が一番頻繁に行っているのは草の根で、「小田原城下町ネット」というところですね。まあ、今回の本ではDEがそこから転載されてきているんですけれど。そこは、ほとんど毎日行っています。

注●小田原城下町ネット：小田原中心の草の根ネット。Sys.op.は、本書にも収録されているDEの作者、BAZUさん。神奈川でも老舗に入る古いネット。回線は9600bps 2回線。

あとNIFTYは、だいたい2日から3日に1回は確実に行くようにしています。まあ、自分のプログラムのアップロードもあれば、RTでネタ探しをやっているときもあるので。あと、ちょっと私は申し訳ないのが梁山泊で。せっかくIDいただいているんですけれども、やっぱりちょっと遠いというのがあって、1週間に1回ぐらいしか行けないんですよね。あとは、やっぱり1週間に1回ぐらいの割合でPEKINへ。そのくらいが主なところですね。他にもいくつか行っているんですけれど、そこは星野美季名義で行っていないので、ここでは内緒ということで（笑)。だいたいそんな感じですね。

たまにしか行かないネットに久し振りに行ったら、アップしていない自分のプログラムがアップされていて、あれには、まいった（YuNK）

注●PEKIN：ぺきんネット。五反田にホストがある、X68000中心のパワーユーザが多いネット。3回線で運営されている。

**Ext**　私は、今はNIFTYだけです。NIFTYで、毎晩、アミを張っていますから（笑)。

**郁**　山猫さんは、先ほど言われた6ヵ所ですか。

**山猫**　IDいただいていれば、けっこう行きますけれど。一番パソコン通信にハマっているタイプでして。電話代も一番多く使っているくらいですから。毎日行くというのは、やはりNIFTYで、メイルの確認というのが多いんですね。メイルをいただくことがひじょうに多いので。あとはMAXとサンデー。ほとんどメイルの確認に回っているような感じですね。梁山泊が週1回くらい。最近、MAXもなかなか入れなくて。1週間に1回、入れたらいいほうですけど。

## 私のアップロードのしかた

**郁**　先ほどYuNKさんが、中途半端なバージョンをアップする場合はどうしてもためらってしまうということを言われていましたけれど。そういう感覚というのは、みなさん、もうかなりのベテランだから忘れちゃいましたか？　最初の頃だけですか？　こんなプログラムをアップして、何を言われるかわからないとかという、怖さみたいなものってなかったですか？

**Ext**　私なんかもう、心臓に毛が生えていると言われてる(笑)。まあ、アップするときは、一応ジャン

クに送ると。そんな感じで、いつでも消せるようにしておいて。まあ、なんでもかんでも全国ネット版に置いちゃいますね。

**美季**　私の場合、さっき言ったようにアセンブラで作る小物類はあまり大手ネットにはアップしないんです。理由は単純で、その手のものって、ほとんどがくだらないものが多くて、たぶん、誰もが1個は持っているだろうというものがほとんどですから。だから、大手にアップする場合、要するに商用ネット、つまり私の場合だとNIFTYなんですが、特にライブラリにアップするものは、ある程度まわりの人間の犠牲の上に成り立っている（笑）感じでして。

でも、基本的にはバグのない安定したバージョンをアップするようにはしているんです。だから、今、NIFTYにアップしているものの場合だと、独立系ではTMSIOと、Mikiが少し古いバージョンですが、ありますね。それとBPLですか。NIFTYでは、あれが一番ダウン回数が多いので、ちょっとびっくりしましたけれど。2,000人以上の方がダウンロードしているんですね。

注●BPL：B-Plusプロトコルのレジューム機能までサポートするプログラム。

**Ext**　あれは本当によかった。

**美季**　私の友達の間では当時BPしかなかったんです。でも、BPはレジュームに対応していなかったので、おかげでBPLがウケたようなんですがね。その3つと、あと、他のフォーラムにときどきアップしたりするときがあります。やっぱり、バグが取れたものでね。.

注●BP：B-Plusプロトコルをサポートするプログラム。

そのかわりといってはなんですが、細かいくだらないやつとか、Mikiのロケテスト版といって、Mikiの中でも「試食版」と書かれた版があるんです。その試食版を、私の場合、小田原城下町ネットという草の根に、とにかく転載でもなんでもしてかまわないけれど、これにはバグがどれだけあって、作者がどういう意図のもとに改造したか全然わからないよという、あくまでそういう合意のもとで使って欲しい、ということでアップするわけです。いわば、「無保証の無保証」というやつですね（笑）。

だから、何が起こっても文句は言わないでねと。バグ報告はとってもうれしいけれど、文句は絶対言わないでねという形で、ドキュメントも何もつけずにアップすることがあるんです。これを、私は「ロケテストバージョン」と呼んでいて、要は知り合いに、とにかく使ってもらって、バグ出しのコスト分担をしようというわけです。当然、私だけでは使えない機能もありますし。知り合いには普通に通信に使ってもらって、私もNIFTYなんか毎晩行っていますから、極端なバグはないはずなんですけれど、日頃あんまり使わないような機能にいきなりバグが出たりすることがありますから。だから、ロケテスト版をばらまいて、草の根でバグの報告という形で協力してもらうということがひじょうに多いんです。

**Ext**　最近、BPLのおかげでキャッチホンをつけることができるようになりました。

**美季**　キャッチホンで回線が切れてもレジュームできるし。

**Ext**　けど一方で、アップするときはレジュームできないのが残念だなあ。

注●レジューム：B-Plusは、バイナリファイルの転送プロトコルで、BPL.XがX68000用のこれをサポートしたプログラム。レジューム機能とは、このバイナリファイルの転送中に電話回線が切れた場合（キャッチホンが働いた瞬間、モデムは切れてしまう）、次回「途中からのダウンロード」を自動的に行う機能。

## バグ報告がくると、ドキッとするより……

**郁**　みなさんがプログラムをアップしはじめた最初の頃の話ですが、はじめてバグの報告を受けたときはどんな感じを受けましたか？

**山猫**　私の場合、バグ報告って、けっこう多いんですよ。平気でバグ付きでもなんでも、できたらいいや、でアップしちゃうんです。ただ、いろいろレポートをもらうんですけれど、メイルのタイトルで「不具合レポート」とか「バグレポート」と書かれると、ドキッとするより……。

**YuNK**　ムッときますね。

**山猫**　私の場合、怒っちゃうんですよね。バグってわかるんだったら、あなた、自分で直しなさいという感じがするんです。最近ひじょうに多いのは、ソフト同士のコンフリクト、衝突ですね。しばらくNIFTYで騒いでいたんですが、どうもTMNとZ-

MUSICの相性がよくないので調べていったら、原因はZ-MUSICの中のベクタのチェックが、前のバージョンだと入っていたんですが、今のバージョンにはないらしいんです。シャープのXCのVer.2.1からOPM関係の関数が変わったんですが、その中で呼び出しているOPMのファンクションが一部変わっているらしくて、そのせいでどうも変なアドレスに飛んでいっちゃうようなんです。対応策としては、(TMNで) 音楽を使わないようにすれば大丈夫だという形でドキュメントに入れておいたんですが。そういった形で、どこに原因があるかわからないことがひじょうに多いんです。だから、最近フリーソフトとはいえ、会社で仕事 (バグ対応の仕事) をしている感じというのが多いですね。

注●ソフト同士のコンフリクト：常駐ソフト同士が同じ命令等を同時に乗っ取りに行き、その結果、衝突してしまうこと。
●Z-MUSIC：西川善司氏作のライセンスフリーのミュージックドライバ。OPMDRV.Xの上位コンパチブルである。
●その後、Z-MUSICの新しいバージョンでは、この問題は解消された。

**Ext** 特にNIFTYの場合、よくわかっていないユーザが多くて。自分側に責任があるときも、ソフトのバグとか言って、よくレポートが出てくるんですよね。そういうのは、ちょっとカチッとくるけど。

**YuNK** バグという言葉を安易には使って欲しくないなと。

**Ext** それ、NIFTYに以前出ていましたね。不具合もいやなの？　不具合はいい、というふうな意見もあったけど。

**山猫** それって、確か私の発言ですよね。バグという言葉にはひじょうに抵抗がある。職業柄もあるんですがね。

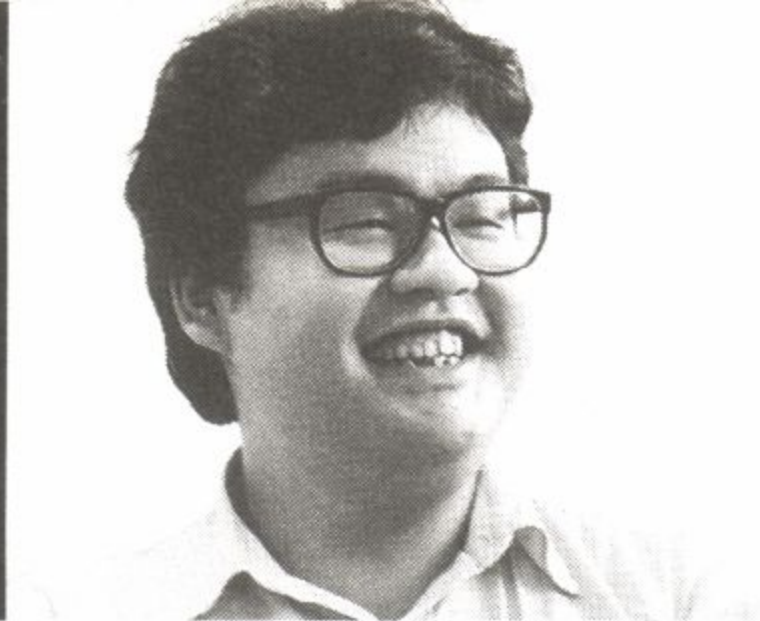
注●山猫さんと美季さんはソフトハウスに勤めている。

**Ext** そう、そう。

**山猫** 仕事でバグっていうのは命取りですからね。そういう事情があるので、バグっていう言葉はやめてくださいと。

**郁** どう言われるのならいいんですか？

**山猫** 素直に、おかしな動作をしたと書いてもらえればね。ここがおかしいと言われれば、「じゃあ、調べましょう」ということになるんですが、いきなり「バグです」って言われちゃうと、「ああーっ」となっちゃう。

バグレポートって書かれると、私の場合、怒っちゃうんですよね。バグってわかるんだったら自分で直しなさいって（山猫）

**美季** まあ、作者のところで動いていることは意外に多いですからね。私なんかでも、Z-MUSICは組み込んでいるんだけれど、他の人のところでは不具合が出て、うちでは出ないというような状況ですから。(そういう報告を聞くと、)「じゃあ、うちは一体、その命令が行ったとき、どのベクタに飛んでいるんだろう」と思うわけです。それを考えると、逆にゾッとしましたけれどね。

でも、やっぱりバグ報告、まあ、プログラムの障害報告とでもいうのかな、要するに、その人にとっての不具合が、たとえば、いきなりバスエラーでも出た日には、その瞬間はその人もカッときているでしょうからね。だから、私はそういうのはしょうがないかなと思うんです。ただ、やっぱり文章で書いちゃうと、あとに残りますからね。だから、ネットに書くときは、カッときたときにはフッと一瞬頭を冷して、CONFIG.SYSからいろいろ抜いてみるとか、AUTOEXEC.BATから何かを抜いてみるとか、いろいろ組み合わせてやってみて欲しいですね。

特に私なんか、常駐ものをやっているので、割り込みの衝突なんて、そこらじゅうで起きていると思うんですよ。最近はなんとなく、どうすれば衝突が起こるのかがわかってきて、割り込み時に衝突させないような手段というのもわかってきましたので、意外と相性がいいのを作りやすくなってはきましたけれど。

それでも、極端な話、TimerCあたりの割り込みを数十個も組み込んだならば、TimerCの割り込みから割り込みの間で全部の処理が終わりきらなかった場合、ものすごい動作になるでしょうからね。そういうことを考えると、やっぱり組み合わせるもののことを考えなければならなくなる。特にフリーソフトというのは、メーカ製のプログラム以上に速度とかを本人たちは重視しているでしょうから。ただやっぱり、組み合わせるプログラムの数が増えてくれば問題も起こってくるでしょうし、そのプログラムがどんなに速度が速くても、他のものがそれについてこれるとは限らないし。ましてメーカ純正のものにもバグがないとは言いきれないわけですし。そう

いうことから考えると、確かに安易にバグ報告と言われても困る気はするけど。

私なんか、はじめてネットにアップして、はじめてトラブルシュートの報告を受けたときは、「あっ、自分で調べる手間が省けた」って、逆に「ラッキー」って思ったものですけどね。

最近は、使う人も多くなっているからトラブルも頻繁になりつつあるのと、その報告を文章にするときに、あまり何も考えずにいきなり文章にしちゃう人がいるみたいですからね。まあ、直接、本人と向きあって話して、相手の顔が笑っていれば冗談めかして言っているんだなとわかりますが、文章だけだと、「何、言ってるんだ、これは」という形になるかもしれないのでね。だから、試用レポートを文章にするときはじっくり考えてから書かないと、えらい目にあうんじゃないですか。

**郁**　やっぱり、みなさん、こういう報告には敏感になっていますか？

**美季**　私はあっけらかんとしちゃいますね。「そうか、動かないのか。でも、こっちでは動いてるよなあ」(笑)。「あとで調べよう」とか思って、すごくあっけらかんと受けるようにしてますけれどね。仕事だけでいいですよ、敏感になるのは。

**郁**　YuNKさんはどうですか？

**YuNK**　ぼくの場合は、もともとけっこう気が小さいもので。だから、プログラムをアップするときも慎重になってしまうということがあるんです。バグ報告がくると、やっぱり「うちでは出てないんだけどなあ」って。「ちょっと申し訳ないんですけど、あなたのマシンのCONFIG.SYSの中身を教えていただけないでしょうか」って、なんとなくこちらが低姿勢になっちゃって。あまり、こうしろ、ああしろってバグ報告に注文をつけすぎてしまっても、バグ報告をくれる側が68について詳しい人間とは限らないわけですから。その結果、バグ報告そのものが減っちゃって、あるはずのバグが見つからないというのではあまりよくないなと思うんです。だから、少々こっちが不機嫌な思いをしても、いっぱいバグ報告が来てくれたほうがまだいいかな、というふうに自分自身で納得はしているんですが。まあ、それでも、やっぱり変な報告だとムッとくるというのはちょっとありますね。

**美季**　人間ですものね。

**山猫**　一番困るのは、しばらくバグが多いバージョンをアップしていたことがあったんです。そうしたら、TMNはバグ付きしかないと思われてしまって(笑)。バグ報告が来なくなったことがありますね。

**美季**　作る側からすると、バグ報告というかトラブル報告って、ひじょうにありがたいものですからね。だから、遠慮なしに問題点を挙げてもらうのはかまわないと思うんですけれど、やっぱり書くときに何か一言欲しいですね。

## 酔っぱらい書き込み 寝ぼけ書き込み

**YuNK**　これは、パソコン通信一般の話にもいえますけれど、オンラインで書くときに思いつきでバーッと書いちゃって、あとで読み返してみて失敗したというようなことってありますよね。特にはじめのうちは電話料金のこともありますし、オフで何を書こうかなとちょっと考えてから書いたらいいんですよね。ぼくも通信で書き込みをするときは、大体オフ書きなんです。オフで書いて、それを見ながら考えて、ちょっとこれではまずいかなと思って消しちゃった、というのがけっこうあるんですけどね。あんまり安易に書くのもなあ、ということはありますね。

注●オフ書き：メイルやボードに書き込む際、通信中にキーを打ってホストに書き込むのではなく、いったん電話回線を切り、ワープロやエディタで文章を書くこと。次回アクセス時にその文章をアップロードすれば通信時間を抑えることができる。

**Ext**　あと、酔っぱらって書くのはいけないね(笑)。

**郁**　経験ありそうですね(笑)。

**山猫**　酔っぱらって書くときもそうですが、通信をやるときっていうのは、大抵夜ですよね。夜、家に帰ってきてから寝る前に書くという……。

**YuNK**　ああ、寝ぼけて書く。

**美季**　寝ぼけて書くっていうのもよくないですね。

**山猫**　あとで自分で読み返してみて、何を書いているんだというのがありますからね。

**郁**　他にも失敗はありますか？

**山猫**　書き込みではないんですけれど、RTやって

て寝ぼけて何を書いているんだか、訳がわからない。
**美季**　話の内容が……。
**山猫**　全然飛んでっちゃった。
**Ext**　寝ぼけてというか、そのまま寝ちゃったというのはよくあるけど。
**山猫**　寝ちゃった人もいますね。
**美季**　やっぱり小さい「ぃ」から始まる人もやるね。

注●小さい「ぃ」から始まる人：会場で傍聴していたicam氏のこと。RTでは「ぃかむ」という名前を使っている。「いくすと」氏の「い」と引っかけての発言と思われる。

**Ext**　だから私の場合は、5分たったら回線が切れるようにしてある。

## フリーソフトとのつきあい方

**山猫**　フリーソフトを使う場合の心構えみたいなものが、やっぱり必要だと思うんです。確かにX68000の場合、市販ソフトがないに等しい状態ですよね。今、フリーソフトのジャンルの中で市販ソフトにないものって、けっこう多いんですよね。そういうわけで、どうしてもフリーソフトを使わざるを得ないところで、みなさん使っていると思うんです。けれど、フリーソフトというのは、市販ソフトのように、お金を払って契約のもとに保証されている形ではないわけでして、使う側もある程度のリスクを背負う覚悟で使わなければいけないなと思うんです。要するに、ハイリスク、ハイリターンですね。そういう心構えを持っていないと、フリーソフトは使いこなせないんじゃないかと思うんですよ。
**美季**　私は自分のソフトをアップするときに、コスト分担ばっかり考えているので、そういう意味では、こっちにとっては利があっても、相手にとっては意味がないようなものなので、あまりそういうことは考えてないですけれど。ただ、普通のフリーソフトを使う場合には、ものによっては寄付を募っているものもあるでしょうし、山猫さんみたいに「絵はがきください」というふうに書いているものもあるでしょうし、いろいろなものがありますからね。だから、そういう意味で、使う側もやっぱりある程度コスト分担という意識を持って欲しいですね。どんなソフトでも、仕事の合間をぬって、もしくは眠い時間を削って作っているわけでして。ひじょうに優秀なフリーソフトというのは、特に68の場合は、これらのソフトを抜きにしては68での生活が成り立ちませんしね。私なんか、もろにそうですから。CONFIG.SYS見ても、AUTOEXEC.BAT見ても、メーカ純正のものが1個もないってのがほとんどですから（笑）。

どんなソフトでも、仕事の合間をぬって、もしくは眠い時間を削って作ってるわけでして、コスト分担の意識を持って欲しいですね（美季）

だから、そういう状況下だと、フリーソフトがないと、逆にいえばMikiだってできなかっただろうし。エディタですら、Emacsを使ったりとか、そういうことばっかりですから。エディタからコンパイラから、要するにメーカ純正というのはライブラリとヘッダだけじゃないかと。ヘッダすらも最近はフリーじゃないかと。

注●Emacs：超強力なエディタ。主にUNIXワークステーションで使われているが、X68000にも移植されている。

**Ext**　あと、HUMAN.SYSね。
**美季**　でも、HUMAN.SYSもパッチ当てちゃってるから。純正のままで使っているものって、ほとんどないわけで。そういうものを使っている立場から言えば、一番気にかかるのは、何と何を組み合わせたら、どうおかしくなったかということでね。他の人もおかしいことがあったら、自分で調べられる限り調べて作者に連絡するとかね。もしくは作者にどうしても連絡がとれないような状況でも、とりあえずボードに書きさえすれば、いつかは作者に伝わるでしょうから、そういったところに書くとかね。

逆に作者とすれば、自分のプログラムに対するレスがそういう形でアップされてくれば、それだけデバッグが楽になるわけですから。だから、そういうコスト分担意識というのを持ってやれば、フリーソフトというのは、どんどん信頼性が高くなって、市販ソフト以上のものになり得るでしょうからね。

GCCなんかはすでにそうですし、安定性という点でいえばEmacsもそうですね。メーカ純正のソフトに比べたって十分信頼性があるわけですし。バグが見つかっても、まあ、1日とか2日というレベルでアップデートされてくるので、メーカ製のようにバージョンアップされるのを黙って待つ必要がないわ

けです。そういうすごい利点がありますからね。ネット上でリアルタイムにどんどん進行できるという利点をもっと生かしてやるべきだと思うんですよね。

**郁** Extさんは、どうですか。

**Ext** みなさんの言われたとおりです（笑）。

## ドキュメントを読まずにいきなりリターン……

**郁** よくフリーソフトを使おうとして、付属のドキュメントを読まずに、いきなり.Xファイルにリターンをかける人がいますね。

**Ext** それ、私ですね。

**美季** 私も、そうかもしれない。

**郁** それで、動かなくて文句を言ってくる人がいるようですが。

**山猫** 私の場合、逆のレスをもらったことがあるんですよね。頻繁にバージョンアップをしたら、バージョンアップのたびにドキュメントを印刷しているので、バージョンアップをやめて欲しいと（笑）。

**美季** だから、私の場合は、試食版にはアップデートのドキュメントだけをつけるようにしているんですよね。どこが変わったかだけをとりあえず教えておいて。あとは、まあ、実行してみればわかるんじゃない、というノリで。

**YuNK** わかりやすいドキュメントを書くというのも、けっこう難しいですよね。

**美季** 私にとっては、プログラムを作るより難しいです。

**YuNK** そうですね。

**Ext** 私のドキュメントがわかりにくいというのは有名だから（笑）。

**美季** だから、最終形としては、なるべくドキュメントを読まなくても、起動すればとにかくわかるようにしてあるのが理想ですよ。とにかく立ち上げれば、なんとなくわかると。

そういう意味ではMacのソフトって有利ですよね。起動すれば、大体わかりますからね。ドキュメントを見なくても、ダブルクリックしてみれば、あとはウィンドウの操作はどれも同じだろうって推測がつきますから。そういうのってX68000のほうならば、SXのアプリケーションの場合は比較的わかりやすいですよね。ドキュメントを読まなくても、なんとなく使える部分が多いですから。私の場合、ドキュメントを読まないで、いきなりリターンする人ですから。

この前ちょっと困ったのが、あるソフトをダウンしたんです。これって、SX用のソフトだったんですけれど、ネットの書き込みで見たときにはあまりそれらしくなかったんです。だから、「SX用じゃないのかな」と思ってリターンってやったら、フッといきなりSXの画面になって。要するに、SXKERNEL.Xか何かが立ち上がってきたのかな。とにかくSXモードで立ち上げちゃって、終われない（笑）。

注●SXKERNEL.X：SX-WINDOW用アプリケーションを単独で起動するときに使用されるプログラム。SX-WINDOWの背景しか表示されない。

で、立ち上がったアプリケーションは画面のどこかに行っちゃっているし。どうもこっちの端のほうにいるらしいんだけど、ADJUST.Rが入ってないから右に行けない。すごく困った記憶がありますね。やっぱり、ドキュメントは読んだほうがいいなと思いますね。

注●ADJUST.R：SX-WINDOWで画面外にカーソルを動かすと、それにあわせて画面全体がスクロールする機能を実現しているソフト。

## ヘルプ機能のつけ方

**郁** あと、フリーソフトを使う場合、“/?”をつけて起動してみるというのもありますね。

**山猫** コマンドラインで使うものの場合は、そういう形で出てくるものが多いんです。たとえば、MikiとかTMNはコマンドラインで使うソフトというより、１つの環境で作っちゃってて、大抵は何もつけずに起動させている。

**美季** 最初の起動も、一応簡易的なヘルプは出しても、それはスイッチの説明だけですからね。

**山猫** コマンドラインで使うツールの類で、ヘルプが出てくるっていうのは絶対実施して欲しいことですね。たとえばDEDITって、出ましたっけ。

**美季** あれ(DEDIT)、使うのにオンラインヘルプっていりますか。感覚的に使いたいな、あれって(笑)。

DEDIT使っていて、ドキュメントを開いた記憶ないものね。

**山猫**　要するに、画面全部使っちゃって環境になっちゃうものは、[HELP]キーを押したら何かしらアクションがあるというのは欲しいなと思うときがある。

**美季**　そう、そう。ドキュメントを読まなくてもいいし。

**Ext**　DEDITって、ヘルプがなくてもわかるでしょう？

**YuNK**　わかります、わかります。

**美季**　あれはメニュー選択形式だから。いつも使うキーは出てるし。[HELP]キー押すと、マクロ記録になった記憶がある。

ドキュメントの最初のほうに、起動と終了の方法だけを書いていただいて、とりあえずそこだけは読んだほうがいいと思いますね。とりあえず終了のさせ方がわからないと、リセットするはめになりますから。

**山猫**　昔、MuTermをはじめてダウンしたとき、（ドキュメントを読まずに）立ち上げたんですが、終わらせ方がわからなくなって。

**YuNK**　あっ、それはやりました。

**山猫**　まさか[SHIFT]＋[BREAK]だとは思わなかった。どうやって終わらせるんだろうって。20分ぐらい悩みました。結局、しょうがないんでリセットかけましたが。

**Ext**　IBM PCのソフトで普通によく使うような部分では、まず最初に終わり方の画面が出て、２回目の起動以降はもう出なくなるとか。そういうアイデアはないですかね。

**美季**　そうですね。WordStarなんかはそういう類ですよね。あれ、最初のうちはずっとメニューモードですからね。本人が慣れてきてキータイプが速くなってくると、どんどんメニューがなくなっている。最後は、メニューも何も出なくなりますから。あれは、最初に使ったときにすごくいいなあと思ったんですが。

**Ext**　それが一番最初か。

**美季**　Emacsのほうは、今、そうなっていますよね。

**Ext**　その手のソフトですね。

**美季**　WordStarは、もう十何年前からそうですからね。だから、あれはすごくよくできているなと思って。でも、自分ではそこまでできないから、とりあえず、ヘルプ機能を充実させるくらいしかないなと。

わかりやすいドキュメントを書くというのも、けっこう難しいですよね（YuNK）

注●WordStar：［CTRL］キーと［E］、［X］、［S］、［D］キーの組み合わせで、カーソルが上下左右に移動をするという、今では常識の１つとなったオペレーションをはじめ、その後の多くのソフトに影響を与えた英文ワープロ。

## ユーザからの要望はどの程度聞き入れます？

**郁**　ちょっと話が前後しますが、ユーザから「こうして欲しい、ああして欲しい」という要望があると思うんですが、取り入れる場合はどういった感じでしていますか。全部取り入れるか、それとも自分が気に入ったところだけ取り入れるのか。

**Ext**　それはもう全部、てんこ盛り。

**美季**　てんこ盛りですか。

**Ext**　TeXのドライバの場合、ユーザはいろいろなプリンタで出力することができます。ところが、私は自分ではそんなにプリンタを持っていないわけで。だからもう、バグレポートもその人にやってもらうしかないわけで、結局、その人が使えるようになるまでつきっきりです。

**美季**　鑑だな（笑）。私は、そこまで言いきれないもの。

**Ext**　ちょっと性質が違うから。

**美季**　確かに、あれは使えないと困る人がいっぱいいるみたいでね。でも、プリンタを買えとは言えませんからね（笑）。じゃあ、こっちのプリンタなら動くから、これを買ってくださいなんて、ちょっと言えないし。

私は要望をもらうと、その要望が10個あったとして、そのうち、実現されるのは３から５個程度ですね。理由は単純で、自分がおもしろいと思ったものだけつけるので。あとは、ちょっと強迫観念があって、バグレポートをいっぱい送ってくれた人の要望

はなんとなく聞いちゃうんですよね。これだけ報告してくれたから、この人の要望は聞いてあげないと悪いかなと。その半面、いきなり要望だけポンと来た場合は、おもしろかったら入るけど、おもしろくないと入らないとか。ひどいときには1年ぐらいたって入ったりする場合がありますから。

Mikiにウィンドウを、というのがそうでしたね。最初の頃からずーっと要望があったんですが、最初のうちはMikiをウィンドウにするメリットがほとんど見つからなかったので、やってなかったんですが、いきなり自分でフッと「こうすれば簡単に窓を開けられるじゃないか」という発想が浮かんだ瞬間に入れちゃったと。そういうのもありますからね。だから、要望というのは、とりあえず出しておいていただければ、いずれは入るかもしれないけれど、やっぱりリアルタイムの反応は半分が限界ですよね。

**Ext** それはメリットよりは、インプリメントの方法がわからなかったからというんじゃないの？

**美季** ウィンドウのシステムに関して言えば、Mikiの場合にはマルチでやる必要はないですから、そんなに複雑なことを考えなくたっていいわけだし。要は、その頃の技術じゃ、ウィンドウの中でスクロールさせたりするのは、それほど速くできなかった。今なら枠で区切られていても、コンソール並みというのは無理だとしても、SXのタイプ.Xとかエディタ.Xよりは速く動かせる自信があるから楽にできますが、昔はそういうのはなかったですからね。やっぱり自分にとって自信のない機能は、ある程度、自信がつくまでインプリメントしたくないというのは確かにありますよね。

**Ext** そういう意味では、そうですね。

**郁** YuNKさんの場合は、どの程度対応していますか？

**YuNK** ぼくの場合はどうかな。一応7、8割ぐらいは聞いているような気がするんですけれど。

**美季** でも、本人の要望が一番多いんじゃないですか、もしかして。

**YuNK** いや、それは、やっぱり納得がいかないと。使っている人にとっては欲しい機能なんだろうけれど、作る側としてはどうも、というところがあって。ぼくも結局、自分で使うために作るというのが第一ですから。だから、ちょっとこれは納得がいかないなと思うと、しぶしぶという感じになっちゃって。この要望は聞き入れられませんと無下に断るのもなんだし。

**美季** まあ、先を楽しみにというぐらいに。

**YuNK** そんな感じですね。「今すぐには無理ですけれど……」みたいな感じでお茶を濁してしまうと。

**美季** 前にMikiに来た追加して欲しい機能というやつなんですが、いくつかのネットに行っているので、それをカーソルかなんかで選べるようにして欲しいという意見が来たんです。TMNには当然そういう機能が入っているんです。Mikiにも欲しいなと言われたんで、しょうがないなと言って。何をやったかというとMMI（Mikiのマクロ言語）で提供したんですね（笑）。とりあえず追加分はマクロでやっちゃえと。で、[OPT.1]＋[Z]へでもバインドしておいてくださいということにしてね。

今、家でもそうなんです。[OPT.1]＋[Z]と押すとペロッとメニューが開いて、カーソルで行きたいネットのところを選択してリターンとやるとオートログインして、オートパイロットまでやっちゃうというやつですから。ここまでマクロファイルでできるんだから、機能を追加しなくてもいいじゃないかと、ごまかしちゃったときがいっぱいありますね。まあ、マクロがついていると、そういう手抜きができます。

**Ext** それは正当だよね（笑）。

**YuNk** それと似たような話ですが、MASMの最近のバージョンでIFとかがありますよね。ブランチに展開されるような。HASのほうで、だいぶ前にあれが欲しいというのがあったんですが、なんとなくMASMのは文法的に好きじゃなかったので対応しなかったんです。あとになって、これはマクロでできないかなと思いついて、今のHASは疑似的に似たような感じの構造化プログラム用のマクロをおまけでつけているんですけれど。あれは、あのおまけを作るためにマクロの機能を拡張したという部分がありまして。で、作ったはいいんですが、ぼくとしてはそういうマクロを書きたいというのが第一にあって、それを使うということは全然考えていなくて、結局自分自身がほとんど使っていないという（笑）。

注●MASM：MS-DOSのマクロアセンブラ。

**美季** で、できたのがハノイですか。

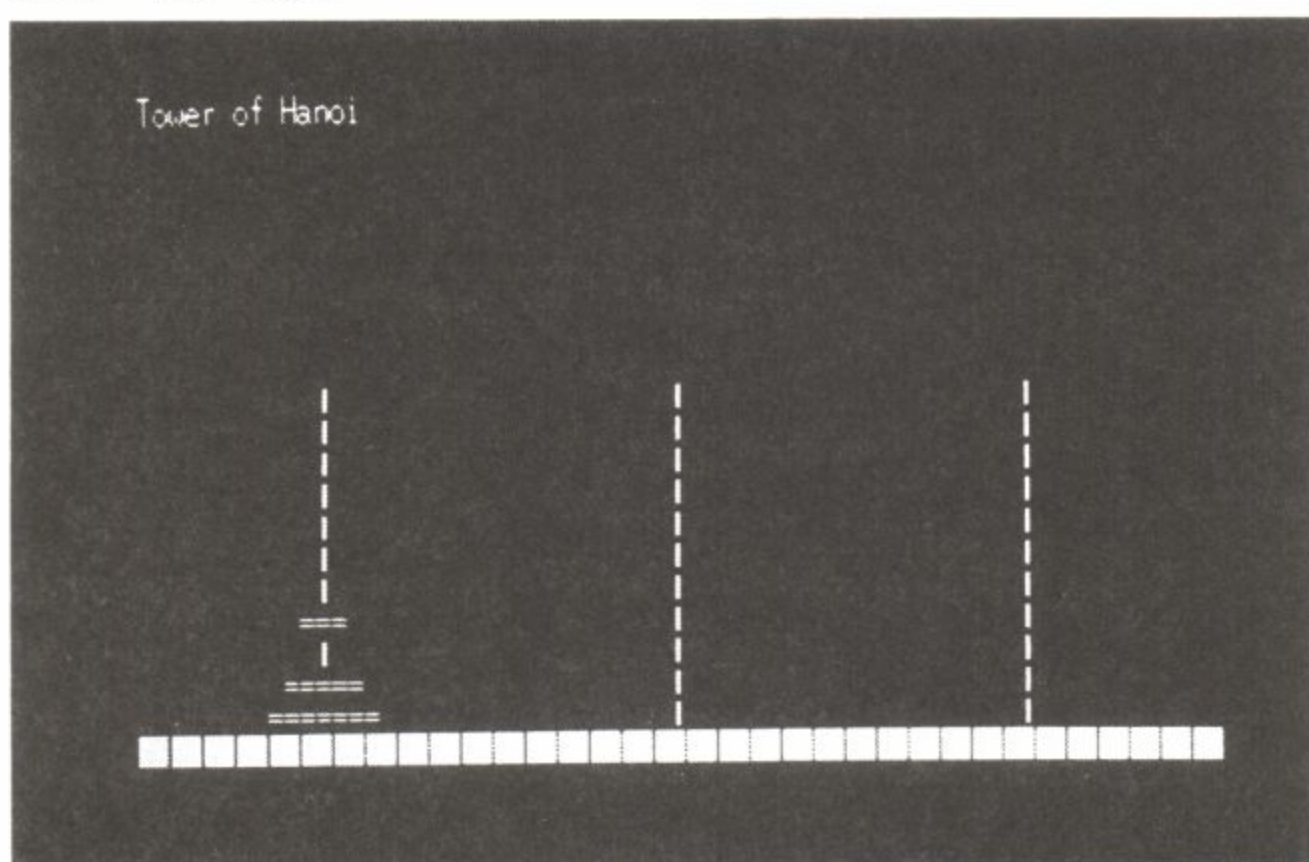
**YuNk** ハノイ、ご存じですか。あれも、そうですね。最初のバージョンのハノイは純正のASでも動く

ユーザからの要望ですか？　それはもう全部、てんこ盛り（Ext）

んですが、マクロの部分のコーディングとかをいろいろ考えていたときに、再帰的にマクロの展開ができるというのがわかっていたので、だったらハノイの塔ぐらいできるだろうと思って。

注●ハノイの塔：数学パズルの一種。3本の棒があって、1本目の棒に大きさの異なる円盤が何枚か大きいものから順にささっている。その円盤を順に動かし、最終的に3本目の棒にすべて移し替えるというもの。移し替える場合は、円盤は一度に1枚ずつしか移動できず、また小さい円盤の上に大きい円盤を重ねてはならないというルールがある。プログラムでは、再帰（自分自身を呼び出すサブルーチン）という手法を使うと簡単に解けるので、再帰プログラムの例として、このパズルが出てくることがよくある。

Pht.2　ハノイの塔

Tower of Hanoi

**美季**　最初、何かと思いましたよ。

**YuNk**　あれは、けっこう笑ったでしょう。最初は、テキストでバーッとハノイの塔を解いていく結果が表示されていくようなマクロを作ったんですけれど、そのうち、もうちょっと凝ってみようと。凝った挙句にエスケープシーケンスでハノイの塔がこうやって。

**美季**　円盤が動いていくと。

**YuNK**　円盤が動いていくという変なものを作ったんですけれど、反響がちょっとイマイチだったんです。あまりにも意味のないもので。

**美季**　あれ、どういう意味を持っているか、わからないんじゃないですか。ハノイを解くプログラムができあがるという変なもの（笑）。

**YuNK**　アセンブラがハノイを解いているというあたりがね。

**美季**　アセンブラがハノイを解いているという、何か変だなと。

**山猫**　TYPEするとは思わなかった。

**YuNK**　あれはちょっとまあ、完全にシャレのつもりでやったんですけれど。

**美季**　私はマクロは好きなので、「そうか、こういう使い方があるのか」って思ったけれど。

**郁**　山猫さんは？

**山猫**　私の場合、作る側の人間である前にMikiユーザだったというのがあるんですね。まあ、言いたい放題、近くにいるものだから言っていたというのがあるんです。だから、相手も同じ気持ちで、やはり言っているんだろうなというのがわかるんです。たとえば、メモ帳を取り入れるときも、何件か欲しいという声があって、最初はメモ帳まではちょっと技術的に作れないなと思ったんですけれど、欲しい、欲しいと言われて。そんなに欲しいと言われると作りたくなっちゃうもので（笑）。

**郁**　押しに弱いタイプですね。

**山猫**　そうですね。要するに、使う側の人間だった頃の記憶があって、要望のうちの6割、7割ぐらいは聞いてしまうんです。あとの4割、3割は、ちょっと今のままじゃできないなと。これをやるためには、一から全部作り直さなければできないなというのがありますので。それと、X68000じゃ、ちょっと無理だぞみたいなのもありますね。

**美季**　めげずにバージョンアップのたびに要望を出すというのが大事だね。毎回言っていれば、いつか入れてくれるから、きっと。1回しか言わないと忘れちゃいますからね。

**山猫**　まあ、うんざりしてその人が作るのをやめちゃうか、入れてくれるかのどっちかですね。

**美季**　私は、いまだにエディタを入れて、と言われてます。最近、入れようかなと思っているんですが。実はバックログのエディットはもうできるようにしちゃっているから、そろそろいいかなと思っているんです。あとは暇ですね。仕事をやっていると、あんまり暇がないんで。

## プログラムにあてる時間はどのくらいですか？

**郁**　みなさん、プログラミングする時間って、1日どのくらいですか。
**Ext**　私は土日だけ。でも、最近、やってないな。
**美季**　私も休日が主ですけれど。場合によっては、休日は日がなキーボードを叩いている日もありますけれど。あと、平日だいたい30分くらい。自分の訓練のためにやっていますね。アセンブラを忘れないように(笑)。一度覚えたテクニックは反復して使うと、あとは目をつぶっていても、そのテクニックが使えるようになるから。だから、常駐方法なんかで「あっ、こういう方法がある」と思いついたら、一時期、似たようなソフトばっかり作るようになりますね。
**郁**　山猫さんは、どうですか。
**山猫**　私も寝る前の30分ですね。まあ、以前、頻繁に機能アップしていた頃というのは、完全に土日つぶしちゃっていたんです。で、私は何のために生きているんだろうと思って（笑)。
　で、そこまでいったら、これ以上続けられないなと思いましたので、平日、それも寝る前に風呂上がりで、頭が乾くまでの時間がありますよね。そういうときに、ちょこちょこと直していって、まとまったところで土曜日の夕方とかにアップするというペースですね。
**YuNK**　ぼくの場合はかなり気分に左右されるほうで。ちょっとノラないなと思ったら、1週間とか半月ぐらいほおっておいたりするんですが、ノってるときはこのへんは学生の特権で、平日でも半徹夜状態になったりとか。土日は1日中とかということもありますね。やっぱり気分によりますね。アセンブラが主体なので、30分ぐらいだと、それまで作っていたプログラムの部分が見えてこないというのがあって、やるならやるで2、3時間は最低でも、という感じになっていますけど。

## 次回作は

**郁**　それでは最後に、次回作というか(笑)、やりたいこととかがありましたら聞かせてください。では、美季さんから。
**美季**　次回作ですか。まあ、Mikiのバージョンアップはこれからも頻繁にやっていく予定なので、これは次回作とはいわないので。ちょっとまだ次の機種が見えていないので、なんともはや、なんですけれど。
　私、今、X68000でデジタルビデオを作っているんです。要するに、ハードディスクに音声と映像を収録していくというのをやっていて、この間、某アニメのオープニングを約2分30秒だか3分ぐらいやって、いきなり10Mバイトのデータが、どん、とできあがったわけです(笑)。で、人にも簡単に渡せないので困っているんです。こういうのを撮ったよと見せたいんですが、ハードディスクごと持っていかないといけないんですね。だから、今度の新しい機種あたりでCPUも速くなるでしょうから、それを考え合わせると、まず、そのオンラインビデオ。究極の目標は、通信でリアルタイムにビデオをやりたいですね。圧縮してでもなんでもいいからね。そうすると、CPUパワーはバカバカ欲しいし。あと、ホストソフトを作りたいです。

注●ホストソフト：通信のホストプログラム。先ほどのWWIVやHOST-PROなどがある。

**郁**　山猫さんは？
**山猫**　ここまで来ちゃうとTMNから抜けられないですから、まあ、そっちのほうは諦めといて。つい、うっかりネットで口がすべってKo-WINDOWとSXにTMNを載っけるぞと言っちゃったことがあるんですね(笑)。いまだに覚えている人がいて、要望が来るんです。そろそろやばいかなと思って。結局、通信ばっかりなんですけれど、そういう形で2つのウィンドウ環境にTMNを載せたいという考えがあって。

注●TMNから抜けられない：これは、座談会時点での発言。その後、TMNはファイナルバージョンがアップされている。
●Ko-WINDOW：X68000のユーザ（小林さんという方）が作

HASを68040対応にすれば、シャープが新マシンを40で出してくれるかも（美季）

ったX68000のためのフリーソフトウェアのウィンドウシステム。

あとは、これも美季さんと昔作った、MZ-2500で動いていたホストプログラムを68に移植して。それをまた自分のホストで使いはじめたんです。そちらのほうのメンテナンスをしながらバージョンアップしたいなと。

あと、X68000から離れちゃうんですが、最近一番いじくっている時間が多いのはHyperCardなんです。下手をするとX68000から離れちゃうかもしれないです。

**YuNK**　ぼくの場合は、とりあえずHIOCSのマウスまわりを安定させるというのが第一なんですけれど。それが終わったら、とりあえずHIOCSのほうは小さいバグ取り程度で打ち止めにしたいなとは思っていますけれど。

**美季**　他のIOCSまわりはどうですか？

**YuNK**　他のIOCSまわりですか。それは要望が出たら、また考えるという感じで。もしかしたら、まだ終われないかもしれないなと。次の機種には、こんなHIOCSなんていう姑息なものは必要としないものであって欲しいなと、切に願うわけですけれど。

アセンブラのほうは次の機種が出るまでに、というわけにはちょっといかないかもしれないけれど、次の機種にあわせて考えたいですね。まあ、早く出てくれることを期待しつつ、上のCPUに対応をしたいなと、今、考えている段階でして。

**美季**　HASを68040対応にすれば、シャープが40で出してくれるかも。

**YuNK**　実は資料だけは集まっているんです。20とか30とか40とかのマニュアルを買いあさっていて。インストラクションコードとかを調べたことは調べたんですけれど、まだ手をつけるに至っていないんです。

**美季**　けっこう面倒くさいコードが多いですからね、20以上は。

**YuNK**　そうでしょうね。アドレッシングモードがとにかく大変なんですね。

**郁**　Extさんの次回作はどうですか。

**Ext**　私は秘密です(笑)。さっきC++とか、BISONとか言いましたよね。あのあたりがカギになりますね。

**郁**　何本くらい予定していますか。

**Ext**　それは、予定だけだったら数十本あるけれど。その中には仕様決定前の段階もあるし。今、コーディングしているのは4本ぐらいじゃないかな。けど、いかんせん、学生のときとは違って土日すらできないような状況ですから。かなり遅れていて、まだまだ先ですね。1つ動いているやつがあるんだけれど(笑)。

**郁**　そういうことを言うと、ネットで突っ込まれますよ。

**Ext**　それじゃあ、お楽しみということで。

**美季**　バグ出しやりますよ(笑)。こっそり回してください。

**郁**　それでは、ずいぶん長くなりましたが、今日はおもしろいお話をたくさんしていただいて、どうもありがとうございました。

# 付録1　BBS一覧

以下にX68000ユーザが比較的多いと思われるBBSを紹介しておきます。取材する時間も十分になく、筆者たちが知っている範囲でコンタクト可能であったBBSに限らせていただきましたので、決して十分なものだとは思っていませんが、最寄りのBBSにアクセスし、あなたのX68000の世界を広げてください。各BBSの紹介文はSys.op.の方（あるいは代理の方）にお願いしました。この場を借りてお礼を申し上げます。

## ●草の根BBS

| BBS名 | 電話番号 | 特　徴 | 入会方法 |
|---|---|---|---|
| 南京<br>（北海道） | 011-757-4001<br>（300/1200/2400bps）<br>011-736-6379<br>（300/1200/2400/4800/9600bps） | アクセスしてもらえればわかるかと思いますが、当ネットの話題はX68000／プログラミング言語／UNIX／アーケード・コンシューマゲーム／AV／銀英伝／マッチメーカというように偏っています。(^^;）会員構成はいわゆる業界の人が多いようなので、そうでない人はちょっと溶け込むのが難しいかもしれない…… | ゲストID：0　パスワードはなし<br>オンラインで必要事項を登録し、問題がなければ、管理者がレベルアップします。ただし、記述内容に不備があった場合は登録を抹消されます |
| SPS-NET<br>（福島県） | 0245-46-1167（代表）<br>Tri-P：SPS | 会員のみなさんが気軽にアクセスできる雰囲気のネットになるよう内容やレスポンスをよくするように頑張っています | オンラインサインアップは行っておりません。封書にて入会の申し込みを行っています。入会金(3090円)、会費無料。なお、詳しい入会方法はゲストアクセスして入会方法の案内を見てください。ゲストID：GUEST、パスワードなし |
| ひるま-ねっと<br>（千葉県） | 0473-30-1347 | 「おきらくごくあく」なネットです。そのうえ、他のネットでは禁止されている行為である「ボードチャット」が公認されています | ゲストID“GUEST”でログイン後、オンラインサインアップをするか否かを聞いてきます。Sys.op.が登録内容を確認したあとに一般会員へのレベルアップを行います |
| サンデーネット<br>（東京都） | 03-3584-7791<br>Tri-P：SUNDAY | 知る人ぞ知るTV番組「パソコンサンデー」の元スタッフが運営しています。そのため、シャープ製品やX68000の情報は充実しています | 正規の入会募集は春のゴールデンウィーク、秋のシルバーウィークの年2回。積極的な書き込み活動をされる方には、そのつど、IDを発行します。ゲストID“sun0000”でログインし、所定の入会申し込み方法でID発行係(sun0002)にメイルしてください |
| 東京BBS<br>（東京都） | 03-3857-9575<br>（300/1200/2400bps、無料回線）<br>03-3857-6211 | 首都圏を代表する50回線総合ネット。X68000関連やホビー、漫画、アニメなど150近いジャンルのボードを用意。CG3000本は圧巻 | 新規会員募集中。ゲストID：NEW。オンラインサインアップで新会員登録 |

| BBS名 | 電話番号 | 特　徴 | 入会方法 |
|---|---|---|---|
| | (4800/9600bps、無料回線)<br>0990-313120<br>(2400bps、有料 1分3円)<br>0990-323892<br>(9600bps、有料 1分6円) | | |
| 梁山泊<br>(東京都) | 03-5698-7733(V.32)<br>03-5698-6800(2400bps、MNP5) | X68000のボードはもちろん、AT互換機、家庭用ゲーム機器関係のボードも充実しています。オフ会も月1回行っています | 新規会員は期間限定(不定期)で募集しています。入会希望の方はゲストID"GUEST" でログインし、住所、氏名、電話番号、希望ID、希望パスワードを記入のうえ、「目安箱」に書き込みを行ってください |
| AFH-NETwork東京<br>(東京都) | 03-5691-0468 | 当ネットは、ゲーマーによるゲーマーのための、ゲーマーのネットです。ゲームが好きな方、大歓迎です | ゲストID：AFH-0000(パスワードなし)ゲストアクセス時にオンラインサインアップ |
| ALTA-NET<br>(東京都) | 03-5474-1871<br>(代表2回線) | X68000中心のハードウェア関係が活発なネットです。X68000ユーザ以外でもハードウェアに興味のある方はぜひ遊びに来てください | 現在オンラインサインアップは行っておりませんが、年数回、前もって告知してオンラインサインアップを行いますので、入会希望者は随時確認のこと。なお、ゲストIDは"ALTA0000"、パスワードはいりません |
| CAT-NET<br>(東京都) | 03-3878-1294<br>Tri-P：CXCAT | 猫を主体とした総合BBS。無料24時間8回線の個人運営で、CG、オンラインソフトなど、さまざまな分野で情報交換を行っています | ゲストID：CAT0000。パスワードなしでアクセスし、トップメニューの"L"コマンドでIDを申請するオンラインサインアップ方式です。IDは1ヵ月以内に郵送で通知されます |
| MAX BBS<br>(東京都) | 03-5707-6720（代表） | オフ会が多くてみんなでよく集まってわいわいやってます | ゲストID"GUEST"でログイン後、オンラインサインアップをするかどうかを聞いてきます。Sys.op.が登録内容を確認したあとに一般会員へのレベルアップを行います |
| PEKIN-NET<br>(東京都) | 03-3447-6852（会員専用）<br>03-3447-6182<br>(ゲスト、会員用) | X68000登場以来の草分け的なBBSですが、昔ほどパワーはなくなっています。むしろ内輪うけに走っております(^_^)； | 新規会員募集は不定期にオンラインサインアップで行っています。約1ヵ月前からネット上で募集の告知をしています。特例でスタッフの認めた方はその場でIDを交付しています。ゲストIDは"GUESTJ"。サインアップはログオフ時に |
| TSUKUMO-NET<br>(東京都) | 非公開 | ツクモフロア内に設置してあるX68000から、各機種の新鮮な情報をお届けします。コンピュータ関連に限らず、お気軽にご利用いただけます | 新規会員随時募集、店頭にて受け付け。これ以外の方法での会員登録は行っていません。登録確認用のIDはGUESTを使用しています |

| BBS名 | 電話番号 | 特　　徴 | 入　会　方　法 |
|---|---|---|---|
| 小田原城下町ネットワーク<br>（神奈川県） | 0465-22-7189（ゲスト用）<br>（24時間/9600bps） | ごく普通のネットでございます。決して裏表があったり、変な話題が中心ではございません。X68000関係については変なフリーソフトを作成したり、集めております | ゲストID：0　パスワード：なし<br>オンラインサインアップ後、数日で正式登録 |
| LDB-NET<br>（京都府） | 075-972-2001 | 多彩なボードがたくさんあり、通信初心者の方も気軽に楽しめるネットです。温かな雰囲気のネットです。年齢層は10～50歳代と幅広く、ボードもコミック、子育て、おちゃらけ等、60ほどあります | ゲストID：GUEST<br>オンラインサインアップできます |
| Naturel（ナチュレ）<br>（京都府） | 075-862-3544 | 多種多様な面々が、得意な分野をフォローして知識を共有していく楽しいネットです。X68000ではメモリボードの自作等も行っています | 新規会員を募集しています。ゲストIDは“GUEST”。BBSボード#1に入会方法が書いてありますので、それに従ってください。週末ごとくらいに発行しています |
| ぺけろく教<br>（大阪府） | 06-878-0082 | 通常のぺけろく教と、気紛れにメンテを行い、「てけろくの教え」というアブナイBBSを開局することがあるのが最大の特徴。93年５～６月頃に大幅なボード、PDSの改編の予定 | ゲストID：X60000　パスワード：guest（半角小文字）<br>ログイン・メッセージに従い、Sys.op.まで必要事項を記入のうえ、メールしてください。ただし、Sys.op.多忙のため、１～３週間ほどID発行に時間がかかることがあります |
| Feel-NET<br>（高知県） | 0888-40-4987 | ホストマシンにX68000を使っています。地方でX68000をさみしく使っている人、仲間は近くにたくさんいるのですよ！ | ゲストID“GUEST”でログイン後にオンラインサインアップ。登録内容の確認後にレベルアップを行います。入会費・会費は無料 |
| Southern-NET<br>（大分県） | 0975-52-4410<br>（2400bps、MNP5） | X68000をホストにして、OS-9のオリジナルBBSソフトで運営。囲碁の対局を目玉にし、OS-9やX68000の情報を交換 | ゲストID：TESTUSER　パスワード：OS-9<br>ログイン後、ボードの#8に入会希望の旨を書き、Sys.op.宛に正式なID、パスワードをメイルで出せばOK!!囲碁愛好者、OS-9に興味のある方、アクセスしてください |

## ●商用BBS

| BBS名 | 電話番号 | 特　　徴 | 入　会　方　法 |
|---|---|---|---|
| NIFTY-Serve | 連絡先　03-3221-7363<br>各アクセスポイントから利用可能 | ニフティが主宰するBBS。全国百数十ヵ所のアクセスポイントから利用可能。シャープUser'sフォーラム（FSHARP）には日本中からX68000ユーザが集う | 書店で「NIFTY-Serveメンバーズパック」を購入し、オンラインサインアップを行うか、所定の書類を事務局に提出する。ゲストでの利用は行えない<br>基本サービス　1分10円 |
| PC-VAN | 連絡先　03-3454-6909<br>各アクセスポイントから利用可能 | 日本電気が主宰するBBS。シャープ系のユーザが集まるところとしては、PC SIG内の「9．コンピュータ／科学（案内1）」内に設けられたX1クラブが中心。談話室やPDSコーナーがある | 所定の書類を事務局に提出するか、専用回線0120-00-9896に電話をかけ、VANPCINFO（大文字・半角）と入力し、番号1を選ぶと、オンラインサインアップが選べる<br>課金は従量制（3分20円）か、固定制（月2000円）を選べるが、サービス内容が異なる |
| アスキーネット | 連絡先　03-3486-9661<br>各アクセスポイントから利用可能 | アスキーが主宰するBBS。X68000関係のSIGとしては、comp.x68kとouter x68k.confer.mainがある。また、SIGではないが、X68000のフリーソフトを集めたpool sharpがある | パソコンショップや書店で「スタータキット」を購入するか、所定の書類を事務局に提出する<br>基本サービス　月2000円 |
| 日経MIX | 連絡先　03-5696-1111<br>遠距離から利用する場合は、DDX-TPやTYMPAS、Tri-Pも利用できる | 日経BP社が主宰するBBS。ソフトハウス、ハードウェアメーカなどが主宰するベンダー会議も活発。シャープ系のユーザの多い会議は、m.others内の分科会x68000。また、シャープの液晶映像システム事業部が主宰するsharp.xがある | オンラインでのサインアップはできない。所定の申し込み用紙を取り寄せて申し込む。<br>基本サービス　970円　月額の上限は1万円 |

**注意**

商用BBSの電話番号は、事務局の電話番号で、アクセス用のものではありません。

# 付録２　参考：X68030での動作について

本書に収録されているフリーソフトウェア('93年２月収録)は、X68000用に開発されたものであり、X68030での動作は保証されておりません。ただ、X68000ユーザとしては、これらのソフトが新マシンでどの程度動作するのかは大変興味があることですし、フリーソフトの作者にとっても、大いに興味がある点だと思います。そこで、グループ68Kでは、ここに独自に動作テストした結果を報告しておきます。

なお、チェックを行ったマシンが出荷前のマシンであったことと、テストする時間があまりなかったこともあって、十分なチェックが行われたとはいえません。とりあえず収録プログラムが起動するかどうかと、主な機能の確認しかしておりません。「動作する」となっているソフトでも、なんらかの不具合が発生するかもしれませんので、ご注意ください。逆に、「動作しない」となっているソフトでも、すぐにX68030対応のバージョンアップ（この点は小回りのきくフリーソフトの強み）が行われると予想されます。したがって、ここに示すのは、あくまでも参考程度の確認であることを最初にお断りしておきます。

**ターゲットマシン**

本体：X68030
メモリ：12Mバイト　コプロ内蔵
5インチFD＋ハードディスク内蔵
OS：Human68k ver3.00

| | |
|---|---|
| ○ | 動作する |
| △ | 動作するが、一部に不具合あり |
| × | 動作しない |
| － | 未確認 |

| ソフト名 | 動作 | 備　考 |
|---|---|---|
| TMN | △ | 通信画面、バックログ画面で、スクロール時に画面にゴミが残る。通信自体は可能。キャッシュオフ注1で画面表示は正常になる |
| MuTerm | × | 画面暗転して起動しない（コンパチモードでも同じ） |
| ish | ○ | |
| LHA | ○ | |
| Bdif/Bup | ○ | |
| Fu | ○ | |
| MF | ○ | |
| LZX | ○ | |
| SEE | ○ | |
| DC | ○ | format-autoで不具合が発生するときは、HEAD-LOADの調整か、FORMAT-ONにする |
| SuperED | △ | 画面スクロールや画面書き換えで、画面上にゴミが残る。編集自体は可能。キャッシュオフ注1にしても、画面上に若干ゴミが残る |
| tsort | ○ | Human68k ver.3拡張機能注2での動作は未確認 |
| dedit | ○ | |
| SRAMCLR | － | |
| TwentyOne | × | HUMAN.SYSのバージョンに依存するので起動しない |
| hcommand | × | COMMAND.Xのバージョンに依存するので差分適用できない |
| caps | × | HUMAN.SYSのバージョンチェックで起動しない |
| float2p | × | |
| HIOCS | △ | スクロール時に画面にゴミが残る<br>キャッシュオフ注1で画面表示は正常になる |

| ソフト名 | 動作 | 備　考 |
|---|---|---|
| FLEXDISK | ○ | |
| DCACHE2 | △ | HUMAN.SYSのバージョンチェックで起動しない<br>バージョンチェックを無視するオプションにより実行可能 |
| DE | ○ | FORMATのバージョンによっては不具合が出る可能性あり |
| ADDDRV | ○ | |
| C/DINIT | ○ | |

注：

1）68EC030の内部に、キャッシュ（命令キャッシュとデータキャッシュが、それぞれ256バイトある）をコマンドラインからオン／オフする命令が追加されている。“cache off[CR]”でキャッシュがオフになる。

2）ファイル名の21文字識別などの機能を使用することができるようになっている。fastopenというコマンドで指定する。

**●郁(かおる)**……………執筆担当：MuTerm、SuperED、tsort

こんなにもコンピュータを知らない私なんかが、このような原稿を書いていることを、まず、お詫びします。しかし、その分、初心者の方向けの文章になっているのではないかとも思っています。ただ、「マスター」の方にはじれったい文章になっていることでしょうけど。なかなか兼ね合いが難しいですね。はぁ。

それから、いろいろお世話になった方々、ありがとうございました。やはり、一番迷惑をおかけしたのは編集の麻生さんでしょうね、ありがとうございました。

私は、「おきらくごくあく!!」なもんで、こんなかるーい文章がいつもの私の口調なので、原稿の口調は長く、苦しいものとなってます。

最後に、逃げも隠れもしません。どこかのネットで見かけたら声をかけてくださいね。

**●静谷　紀方**……………執筆担当：第Ｉ章、ish、LHA

いやぁ、終わった、終わった（まだ終わってないって←よくあるオチですいません(^^；)。自分の力量も考えずに膨大な量の（私にとっては膨大だった（笑））原稿をうっかり引き受けてしまったのが運のつき。担当編集者の方をはじめ、みなさんに迷惑かけまくりの私でした。ほんとにどうもすいませんでした。結局、時間その他の限界で、書きたかったこと、書かなければならなかったことが全部書けたとは到底思えませんが（ほんとはこのあとがきも「チャット形式にしよう」とか、そういう企画もあったんだよね）、私の駄文を読んで（あ、「駄文」なのは私のとこだけよ（笑）、）１人でも多くの人がパソコン通信にハマってくれればなぁ、と思っています。それこそが私がこの原稿を引き受けた理由なのですから……。みんなぁ、ネットで待ってるからねぇ（笑）。

**●白井　一義**……………執筆担当：ADDDRV、DCACHE2、DE、HIOCS、SRAMCLR、TMN

編集の方はもちろん、まわりに迷惑かけっぱなしの半年間でしたが、なんとか終わりました。血の匂いこそしませんが(某誌から引っ張ってきたネタ(笑))。寝ているはずだった時間や、数十箱のタバコと数知れないコーヒーを費やした結果がこれです。ネットワーカ諸氏から見れば、いろいろご意見もありましょうが、笑って許してお読みください。

私の場合、最初にこの本の原稿を書かないかと話が回ってきたときは、「絶対にやらん！うんなもん、苦労するだけやんか！」と断言していたのですが、いつのまにやら巻き込まれてしまいました(実際苦労しましたけど(^_^；))。おかげでよい経験をさせてもらいましたが、心残りなのは、事情によって収録できなかったプログラムがあることと、この本が出るのは噂の新機種が出てしまったあとになるであろうこと。さて、いくつ動くかな？

●馬場　雅嗣……………執筆担当：DC、FLEXDISK、Bdif & Bup、MF

念願かなって、初物を食べた感じですね。私はこのような正式な原稿を書いた経験もなく、言い回しが未熟なことを自覚しています。実は原稿を書くのと並行して自分の作成していたプログラムのバージョンアップを行っていまして、書きはじめてからこの本に収録されるまでにバージョンは2つ上がっていたりします。これは、他の方の作ったプログラムにも言えまして、中には書きはじめた時点ではテストバージョンだったのが、あれよあれよという間に公開して、なんてのもありました。しかし、自分が作ったプログラムの解説ほどいやなものはないですね、あんまり褒めても何か言われそうだし、かといって、けなすわけにもいきませんから。いやー、本当にMF.Xだけは他の人に書いて欲しかった。みんな嫌がったけど。そもそもMF.Xを入れる都合上、私に執筆のおはちが回ったわけですが。

●微……………執筆担当：SEE、dedit、TwentyOne、hcommand、LZX（リライト）

思えば、X68000は、貧困学生だった頃、借金して買った夢のマシン。その後、就職で上京、ネットに参加し、PDSに感動し、そして最後に、この本の執筆に参加することができたのもX68000のおかげです。

老舗ネット・パワーユーザの方々をさしおいてという心苦しさはありますが、自分のX68000ライフの最後の記念になりました。今や妻も子もある身になり、いつまでもあの頃のように、好き勝手にやっているわけにはいかなくなってきました。そろそろ私もX68000から卒業です。万感の思いを込めて。さようなら、X68000。

なんて言っておきながら、まだ値段も決まってないのに、昨日、予約を入れてきました。許せ千花子、泣くな和輝。さて、いくらになったのやら、私のX68030。

●Ya.O……………執筆担当：float2p、LZX、Fu、C/DINIT、CAPS

「ふだん自分で使っているものばかりだし、ドキュメント読めば、使用法はすぐ書けるね」と気軽に引き受けたのが間違いのもと。「ゲッ、環境設定が必要なものばかりだ。」「やっぱ、機能一覧じゃ読んでておもしろくないからヤダナ。」そして、出てくる、出てくる、ふだん使っていなかった機能。おまけに、パソコンに触れる時間がどんどん短くなってゆく。5000円の「テレジョーズ」の固定料金割っちゃってる。おまけに家の電源は不安定で、落ちる、落ちる。壊れるプログラム、辞書、そして書きかけの原稿（泣）。原稿が遅れて、みなさんにご迷惑おかけしました。本当にすみません。その分、いつもと違う態度でフリーソフトウェアに接することができた気がします。「どうやってんだろ？」ってDISしたり、変な設定で動かしてみたり……（そのわりに原稿に反映されてないけど）。

## R



## S



## T



## U



## V



## W



## X



## Z

# INDEX ●和文

## す



## せ



## た



## ち



## て



## と



## は



## ひ



## ふ



## ほ



## ま



## み



## も

## ゆ



## り



## れ



## ろ

# X68000 Free Software Book

1993年3月25日　初版第1刷印刷

1993年3月31日　初版第1刷発行

編　者…………グループ68k

発行人…………孫　正義

発行所…………ソフトバンク株式会社　出版事業部

〒108　東京都港区高輪2-19-13 NS高輪ビル

営業部　☎03(5488)1360

編集部　☎03(5488)1326

印刷所…………株式会社　厚徳社

組　版…………株式会社　帆風





ISBN4-89052-392-8　C0055

SOFT BANK ソフトバンク

# X68000 Free Software Book

X68000 Free Software Book

グループ68K 編

SOFT BANK

ISBN4-89052-392-8 C0055 P2900E

SOFT BANK ソフトバンク

定価2,900円（本体2816円　フロッピーディスク含む）

# X68000 Free Software Book

●通信関係

| | |
|---|---|
| TMN | ウィンドウを多用し、強力なマクロ言語を備えた高機能ターミナルプログラム |
| MuTerm | コンパクトで高速、高機能なターミナルソフト |

●通信支援関係

| | |
|---|---|
| ish | 通信必携のテキストーバイナリ変換プログラム |
| LHA | 高率圧縮を行う標準アーカイバソフト |
| Bdif & Bup | 通信ユーザ必携のバイナリ差分抽出更新プログラム |

●ファイラ

| | |
|---|---|
| Fu | 便利で使いやすく、自由度の高いファイルユーティリティ |
| MF | ユーザが独自の環境を構築できる、柔軟性に富んだ2画面型ファイラ |

●ツール

| | |
|---|---|
| LZX | .X,.Rファイルを実行可能なまま圧縮するツール |
| SEE | LZH（LHA）ファイルの内容も確認できるファイルビューワ |
| DC | 超高速、連続複写可能なディスクコピーツール |
| SuperED | 超高速、高機能なED.Xコンパチブルエディタ |
| tsort | 数字を認識できるディレクトリsortコマンド |
| dedit | X68000ユーザ必携の多機能ディスクエディタ |
| SRAMCLR | 内蔵SRAMの内容を必要に応じて消去するプログラム |

●システム関係

| | |
|---|---|
| TwentyOne | パワーユーザ御用達のシステム強化プログラム |
| hcommand | TwentyOneなどのシステム拡張機能に対応させたCOMMAND.X |
| caps | プログラムごとに自由にキー割り当てを変更、拡張できる常駐プログラム |
| float2p | 純正float2.X (Ver.2.01)をさらに高速化した浮動小数点演算ドライバ |
| HIOCS | 高速文字表示が可能なコンソールドライバ |
| FLEXDISK | 高速、再確保可能RAMディスクドライバ |
| DCACHE2 | デバイスドライバに割り込んで動作するディスクキャッシュプログラム |
| DE | ドライブ名をデバイスごとに設定するツール |
| ADDDRV | デバイスドライバ用ユーティリティ |
| C/DINIT | 起動時にCONFIG.SYSを選択することができるデバイスドライバ |

X68000 Free Software Book

グループ68k……編

SOFT BANK